Australian Government
Department of Defence
Defence Science and
Technology Organisation

Modelling Agent-Environment Interaction in Multi-Agent Simulations with Affordances

Michael Papasimeon

Air Operations Division

Defence Science and Technology Organisation

DSTO–RR–0349

ABSTRACT

A computational model of agent-environment interaction is presented that has been inspired by the theory of affordances from ecological psychology. The model is developed in the context of agent-environment interaction in multi-agent simulation in a virtual environment. The research is motivated by multi-agent simulation of military operations, where the agents represent cognitive models of military operators. A mechanism for evaluating the model is provided through a number of design and implementation interpretations in two distinct large scale multi-agent simulation systems.

Published by

DSTO Defence Science and Technology Organisation
506 Lorimer St,
Fishermans Bend, Victoria 3207, Australia

Telephone:	*(03) 9626 7000*
Facsimile:	*(03) 9626 7999*

Preface

Some of the ideas described in this report relating to the use of affordances in intelligent environments that are suitable and accessible to computational agents were first presented [111] at the Australian Cognitive Science Conference in 2002 (First year of my part-time PhD candidature).

- M. Papasimeon. Intelligent Environments for Agents, *2002 Australian Cognitive Science Conference (OzCogSci '02)*, Fremantle, Western Australia, Australia.

As part of the research, a prototype multi-agent military simulation has been developed, named Human Agent Virtual Environment (HAVE). The development of the HAVE multi-agent simulation is described in Chapter 2 and was originally published as the following conference paper in the AAMAS conference in 2007.

- M. Papasimeon, A. R. Pearce, Simon Goss, Clint Heinze and Tim Patterson. The Human Agent Virtual Environment, *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2007)*, pp. 108, Honolulu, Hawaii, USA, 2007.

Elements of the research have also appeared in other refereed publications as follows.

- Clint Heinze, Michael Papasimeon, Simon Goss, Martin Cross, and Russell Connell. Simulating Fighter Pilots. In Simon G. Thompson Michal Pechoucek and Holger Voos, editors, *Defence Industry Applications of Autonomous Agents and Multi-Agent Systems*, Whitestein Series in Software Agent Technologies and Autonomic Computing. Birkh auser Basel, 2008.
- M. Papasimeon and C. Heinze. Extending the UML for Designing Jack Agents, *2001 Australian Software Engineering Conference (ASWEC)*, pp 89–97, 2001
- S. Goss and C. Heinze and M. Papasimeon and A. R. Pearce and L. Sterling. The Importance of Being Purposive: Towards Reuse in Agent Oriented Information Systems *Workshop on Agent Oriented Information Systems (AOIS 2003)*, Melbourne, Australia. P. Giorgini and B. Henderson-Sellers and M. Winikoff (eds), Lecture Notes in Artificial Intelligence (LNAI), pp. 110–125, 2004
- C. Heinze, M. Papasimeon, and S Goss. Issues in Modelling Sensor Fusion in Agent Based Simulation of Air Operations. In *Proceedings of the Sixth International Conference on Information Fusion (FUSION '03)*, pp 296-301, Cairns, Australia, 2003.
- C. Heinze and M. Cross and S. Goss and T. Josefsson and I. Lloyd and G. Murray and M. Papasimeon and M. Turner. Agents of Change: The Impact of Intelligent Agent Technology on the Analysis of Air Operations, *Advances in Intelligent Systems for Defence*, Edition 1, L. Jain, N. Ichalkaranje and G. Tonfoni (eds), pp 229-264, World Scientific, Series on Innovative Intelligence, Vol. 2, 2002

- C. Heinze and M. Papasimeon and S. Goss. Specifying Agent Behavior with Use Cases, In Proceedings of Pacific Rim Internation International Workshop on Multi-Agents (PRIMA 2000), pp. 128–142, 2000

Modelling Agent-Environment Interaction in Multi-Agent Simulations with Affordances

Executive Summary

This research report is a reproduction of Dr. Michael Papasimeon's PhD thesis *Modelling Agent Environment Interaction in Multi-Agent Simulations with Affordances.* The thesis was undertaken in the Department of Computer Science and Software Engineering at the University of Melbourne. The research that led to this thesis was supported by DSTO from February 2002 to February 2009 under the Air Operations Division Long Range Research program. Specifically the research was conducted in the Air Operations Research (AOR) branch LRR task (currently LRR 07/245). The research was supervised by Dr. Simon Goss in DSTO and Dr. Adrian Pearce from the University of Melbourne. Professor Leon Sterling and Professor Liz Sonenberg were members of the advisory panel for the thesis. In order to make the research more easily accessible to the DSTO community it is being republished in a DSTO format.

The research has been undertaken as part of a larger research program into intelligent agents and multi-agent systems going back to the early 1990s. The research program aims to better model the situational awareness of computational representations of fighter pilots in military simulations. These simulations are used to conduct operations research studies, the results of which are used to provide advice to the Australian Defence Force (and in particular the Royal Australian Air Force). This advice is used to make decisions on the acquisition of new military aircraft and to develop new tactical procedures and concepts of operations for aircraft in service.

This thesis presents a new computational model of agent – environment interaction that has been inspired by the theory of affordances from ecological psychology — the theory that humans and animals can directly perceive the action possibilities of the world around them. The aim of this work is to provide new approaches to the design of the interaction mechanism between agents and the environments in which they are located. An approach inspired by affordance theory can allow for more situated, richer and complex agent behaviours in a virtual environment.

The scope of the work is multi-agent simulation of military operations in particular air combat operations, where the agents represent cognitive models of military operators such as fighter pilots. The research is motivated by the need for humans (for example pilots) and agents (for example simulated pilots or environmental entities) to interact within a shared virtual environment on an equal footing.

The problem with current agent programming techniques typically utilised is that they do not necessarily capture the salient agent – environment interactions. Existing approaches to developing richer and more complex intelligent agent behaviours have focused on developing (i) a more intelligent environment or (ii) a more intelligent agent. This thesis shows an alternative approach, one where the focus is on the interaction between the agent and the environment that has the potential for more situated agent behaviour.

Specifically, the interaction is represented using a computational model of interaction inspired by affordance theory. The model allows for affordances or action possibilities for

each agent to be computed in a manner which allows for flexibility in the design of the multi-agent simulation. The model captures the key characteristics of the affordance concept which are relevant to military multi-agent simulations. These include the intentional and relational nature of affordances as well as their ability to be directly perceived by agents in the environment.

The model is implemented in two complex multi-agent simulations. One a large scale real world military multi-agent simulation, the other a smaller implementation of an adversarial game. These implementations allow for the evaluation of the model in a qualitative and quantitative manner. Consequently, the main contributions are:

- An affordance based computational model of agent – environment interaction, and
- an analysis of the impact of adopting an affordance based approach to agent – environment interaction in multi-agent simulations.

The evaluation addresses the impact of an affordance based approach from a number of different areas. It includes an assessment of the impact on the design of multi-agent simulations when an affordance based approach is adopted. Specifically it looks at two cases, affordance computation in the environment and affordance computation in the agent. The evaluation also looks at the impact an affordance based approach has on the computational performance of a multi-agent simulation.

Author

Michael Papasimeon
Air Operations Division

Michael Papasimeon graduated from the University of Melbourne in 1999 with a Bachelor of Science (Honours) in Physics and a Bachelor of Engineering (Honours) in Software Engineering. In 2009 he completed a Doctor of Philosophy in Computer Science, also from the University of Melbourne.

He currently works as an operations analyst in DSTO's Air Operations Division. His professional experience has been focused on research, modelling, simulation and analysis of air combat operations.

Contents

Figures

Tables

Chapter 1

Introduction

"The meaning or value of a thing consists of what it affords."
— James J. Gibson [1]

1.1 Introduction

This thesis is concerned with modelling the interaction between computational agents and virtual environments in multi-agent simulations. Specifically the approach taken to modelling this interaction is inspired by the theory of affordances [52]; the theory that humans and animals not only perceive the properties of entities in the environment, but also the meaningful action possibilities associated with these entities.

The theory of affordances belongs to the field of ecological psychology, the study of how humans and animals interact with the real environment. Multi-agent simulations are typically computational models of real world systems. Hence, an understanding of how humans interact with the real environment can provide insights into the design of the interaction of computational agents and virtual environments inside a multi-agent simulation.

While taking inspiration from affordance theory, the purpose of this work is not to develop a psychological model of affordance. Rather, concepts and ideas from affordance theory are used to provide guidance and insight into how the interaction between agent and environment in multi-agent systems can be better designed. This type of approach promises to allow for the capture and representation of richer and more authentic intelligent behaviours in multi-agent simulations.

A richer interaction is consistent with the view from the field of situated cognition [26, 146] that complex behaviour arises from the interaction between an agent [2] and the environment rather than the disembodied reasoning of an isolated agent. A richer interaction with the environment is also meaningful, dynamic, context sensitive, relational, action oriented and tailored to the individual agent – all characteristics of affordances.

[1] James J. Gibson developed the theory of affordances – the idea that humans and animals perceive meaningful action possibilities in the environment.

[2] Here the word agent is used in its most general sense and refers to any intelligent actor situated in and interacting with an environment, whether it be human, animal or an intelligent computational entity.

Existing approaches to capturing more intelligent behaviour in multi-agent simulations have typically involved

1. designing a more intelligent agent, or
2. designing a more intelligent environment.

Efforts aimed at designing more intelligent agents have included the development of increasingly sophisticated agent reasoning models [120, 121, 16, 83], abstractions for the social coordination of groups of agents in teams and organisations [60, 139, 154, 155, 152, 147], as well as the development of complex agent ontologies.

At the other end of the spectrum, work in the area of intelligent virtual environments [36, 37] has emphasised their annotation with labels that are accessible and understandable by agents. The important role the environment plays in determining intelligent behaviour in multi-agent systems has been also recognised by the multi-agent systems community [173, 115, 168, 160, 73]. The shifting of intelligence from agent to the virtual environment has the potential to simplify the design of individual agents and also the overall design of a multi-agent simulation.

However, neither of these approaches have provided significant insight into addressing the problem of humans and agents *interacting* within a shared virtual environment on an equal footing. An alternative approach is to focus on the interaction between agent and environment rather than either one in isolation. An interaction based approach is motivated by the understanding that intelligent behaviour arises from the interaction between an agent and its environment. It is this approach that has been adopted in this thesis. Specifically, this work focuses on modelling and understanding the interaction between agent and environment using affordance theory as a design guide.

The theory of affordances was developed by Gibson [53] as a mechanism for describing the interaction between biological cognitive agents (such as humans and animals) and their environment. This theory is one of the most important ideas in ecological psychology. It is central to the concept espoused by Gibson that when one considers the environment from an ecological rather than a purely physical perspective, meaning in the environment is discoverable. That is, agents can perceive the environment in a manner that is meaningful to them. Gibson originally defined affordances [3] as follows.

> *"The affordances of the environment are what it offers the animal, what it provides or furnishes, either for good or ill."*
>
> — James J. Gibson [54]

Since Gibson's inception of this concept, there has been considerable refinement and debate in a number of communities on the definition, nature and application of affordances. [4]

[3]Gibson specifically notes that although the verb *to afford* can be found in the dictionary (in relation to providing an or supplying an opportunity or facility), the noun *affordance* does not appear, stating that he made up the term.

[4]While this debate has been primarily centred around the ecological psychology community, there has been important contributions from a number of different fields including those of industrial design, human computer interaction, situated cognition and multi-agent systems.

Aspects of the debate on the nature of affordances that are relevant to the design of multi-agent simulations are discussed in more detail in Section 5.2.

In this work, affordances will be considered from the perspective of computational agents situated in a virtual environment in the context of a multi-agent simulation. Specifically, this research looks at modelling the perception of the affordances in which the virtual environment provides the agent. The view of affordances that is adopted in this work is one of action possibilities or opportunities for agent action in the environment. Thus in this research, affordances are defined as follows:

Definition 1. *Affordances are the action possibilities an environment provides an agent.*

This definition is quite broad especially when considering the wide range of ideas that can be described by terms such as *agent* and *environment*. However this work looks at modelling the agent-environment interaction using affordances for very specific types of agents and for very specific types of environments.

In this thesis, the focus is on computational intelligent agents that are reasoning models of human decision making. These intelligent agents are situated in a virtual environment representing the real world environment. The agents, the environment in which they are situated and their interaction is represented as a multi-agent simulation.

It is in this type of software system that this research considers affordances as a mechanism for agent-environment interaction. While multi-agent simulation of air combat [135] is the motivating example of this type of software system, this work has potential application to other types of multi-agent simulation ranging from traffic simulation to interactive entertainment such as video games.

The dynamic annotation of virtual environments with affordance (or action possibilities) has the potential to provide a much richer and more interactive experience, not only for the agents situated in these environments but also for the humans that interact with them. As such, this research has the potential to have application beyond the completely virtual environments of multi-agent simulations and video games in fields such as mirror worlds [49] and augmented reality [7].

Virtual worlds such as mirror worlds that encompass completely virtual representations of a physical world, [5] could also benefit from affordances. Mirror worlds are typically encoded with useful information, that could also be designed to provide information about the action possibilities that a person making use of such virtual environments would find useful.

Similarly in the field of augmented reality, where the real world is augmented, supplemented or complemented with additional information generated in a virtual representation, there is the potential for affordances to provide a richer experience augmented reality experience. The prospective applications range from using augmented reality for guidance and navigation,[6] or in military applications such as heads up displays in fighter aircraft which provide pilots with additional action oriented information to help the complete their mission.

[5]Google Earth can be considered a typical example of a mirror world.

[6]For example in a museum or to help a tourist navigate a city

Significant progress has been made over the last two decades in the development of theories, architectures and languages for agent reasoning across a number of communities.[7] However the deployment of large scale and real world multi-agent simulations in fields such as military operations research has highlighted a number of issues [56, 68].

A critical issue in multi-agent simulation is the rise in agent complexity. While this is driven by the desire for increasingly sophisticated agent behaviour it has meant that there has been an increase in complexity in the specification, design, implementation and testing of these systems. The problem is exacerbated by the requirement for coordination with other agents and complex interaction with the environment. To date, managing this complexity has largely been addressed by either the development of new agent reasoning models, the refinement of existing models or through the development of many different agent-oriented software engineering tools and methodologies.

Another significant issue is the role of the environment in many of these agent theories, architectures and languages. While the role of the environment is emphasised in most definitions of agency, its role is significantly less prominent in most agent architectures and languages [171].

As a result, it is important for the designers, programmers and users of multi-agent systems to be able to effectively understand and explain complex agent behaviour; behaviour that is not only driven by the agent's own internal mental state but also the agent's interaction with the environment.

1.2 Agent-Environment Interaction Architectures

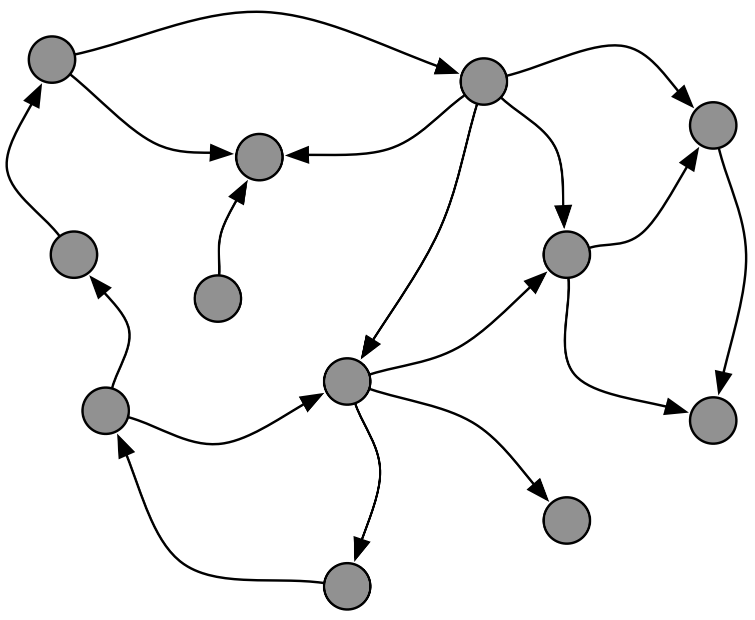
The interaction between an agent and its virtual environment is important to many application domains. However, many agent theories and systems do not explicitly represent the environment, and those that do, treat the interaction in a rudimentary manner.

For example, many pure multi-agent systems consist of a set of agents which communicate directly with each other via the exchange of messages, as illustrated in Figure 1.1. While such systems are suitable for some types of applications such as modelling social interaction, a lack of an explicit model of the environment makes them less suitable for other domains.

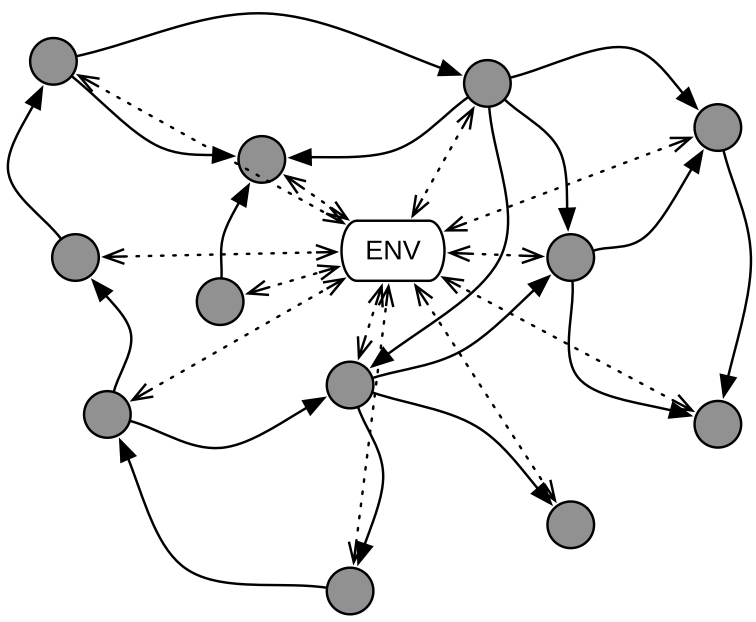
Pure multi-agent systems can be modified through the augmentation of an explicit model of the environment. Two examples of such a hybrid approach are shown in Figure 1.2. In the first example the existing inter-agent architecture remains and interaction with an environmental model occurs on an as-needed basis by each individual agent. In the second example, the environment forms the hub which all agent-environment and agent-agent interaction occurs.

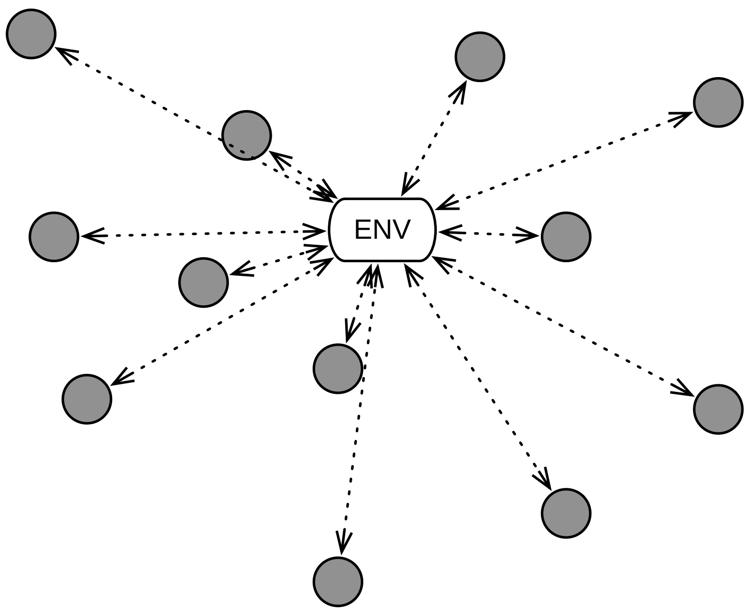
These two approaches characterise possible design solutions when an existing multi-agent system is integrated with an existing legacy environmental system. On occasions when

[7]While most of this research has been centred around the multi-agent systems and artificial intelligence communities, there have been also been contributions from fields such as the simulation and interactive entertainment communities.



Figures 1.1: Pure Multi-Agent System. A conceptual example of a pure multi-agent system represented as a directed graph. The nodes represent the agent in the simulation and the directed arcs represent directed communication links which allow agents to directly communicate with each other via message passing over these links. In this type of pure multi-agent system there is no explicit model of the environment. The environmental model is implicit through the ability of agents to communicate directly with one another via message passing.





Figures 1.2: (Above) Hybrid Multi-Agent System 1. An example of a hybrid multi-agent system where an explicit component representing the virtual environment has been added to the system. Direct communication between agents is still allowed, but other types of interaction occur through the environment. (Below)Hybrid Multi-Agent System 2. An example of a hybrid multi-agent system where all interaction between agents occurs via an explicit representation of the environment. Although the agents in this figure are represented external to the environment, it is preferable to conceptually consider all the agents as part of (or situated in) the virtual environment.

a new multi-agent simulation is being developed, it is preferable to consider the close interaction between required between agent and environment at design time.

Regardless of which approach is taken to integrating intelligent agents with a virtual environment, one must consider the actual nature of the agent-environment interaction in detail. This raises many theoretical, design and implementation issues.

Traditionally, theories of agency have described the interaction between an agent and the environment via the exchange of percepts and actions. Russell and Norvig's environmental simulator [131] is a good example of this type of interaction. This model of agent – environment interaction has been adapted and is presented in Algorithm 1.1.

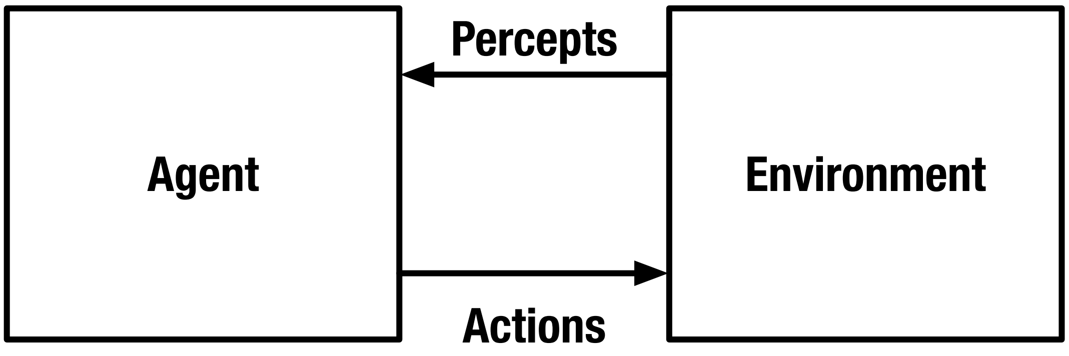
Algorithm 1.1 Russell and Norvig's Environmental Simulator

```
 1: procedure RUN-ENVIRONMENT($state$, UpdateFn, $agents$, $termination$)
 2:     inputs:
 3:     $state$,                          ▷ Initial state of the environment
 4:     UpdateFn,                         ▷ Function to modify the environment
 5:     $agents$,                         ▷ A set of agents in the simulation
 6:     $termination$                     ▷ Predicate to determine when we are done
 7:     repeat
 8:         for all $agent \in agents$ do
 9:             Percept[$agent$] ← GetPercept($agent$, $state$)
10:         end for
11:         for all $agent \in agents$ do
12:             Action[$agent$] ← AgentReasoningProgram[$agent$](Percept[$agent$])
13:         end for
14:         $state$ ← UpdateFn($actions$, $agents$, $state$)
15:     until $termination(state)$
16: end procedure
```

The interaction between agent and environment is represented through a number of key steps. The first step is that for each agent in the environment the function GetPercept is used to determine what each agent can perceive in the environment based on the current environmental state represented by the variable *state*. Each agent's percepts are then processed by each agent program represented in the algorithm by AgentReasoningProgram producing a set of actions. These actions are then undertaken in the environment via the function UpdateFn which updates the current state of the environment.

This represents the most traditional, typical and most widespread model of agent – environment interaction. This model of interaction more commonly illustrated by Figure 1.3, considers agents and environments as two separate components which interact via the sending of percepts from the environment to the agent and subsequently the sending of actions from the agent to the environment.

Although such a description is elegant and abstract, it does not provide guidance as to how these percepts and actions should be represented in a real system. This type of model also does not capture the situated nature of the agent – environment interaction. Furthermore, it treats percepts and actions as distinct and separate concepts. This is in contrast to the view from ecological psychology where the concept of an affordance can be a mechanism



Figures 1.3: Traditional Model of Agent-Environment Interactions

for relating a percept to an action – that is affordances can be considered as directly perceivable action possibilities.

Understanding and modelling the interaction between agent and environment is central for developing sophisticated and intelligent agent behaviours. A number of design options are available when faced with the need to develop such behaviours.

The first option is to modify existing agents to make them more sophisticated and more complex so that they can better process the information being received from the virtual environment. Making an agent more sophisticated may include designing an improved perception module [72], improving an agent's reasoning engine or simply adding to the repertoire of behaviours available to the agent. [8]

One of the advantages of developing agents for completely virtual environments is that in some cases the designer has the ability to influence the design of the virtual environment itself, to make it more accessible and amenable to the agents inhabiting it. This can typically be done by annotating the environment with labels or annotations [91, 39, 37, 36] that the agents can perceive and reason about. These techniques are often categorised as intelligent virtual environments.[9] However, the ability to modify a virtual environment to make it agent friendly is not always possible, especially when dealing with closed legacy virtual environments.

This implies that if one desires more intelligent agent behaviour, this can be achieved by either making the agent more intelligent (for example via a more intelligent reasoning engine) or by making the environment more intelligent (for example by annotating it in an intelligent and meaningful way). As an aside, it raises an interesting question – where does the (artificial) intelligence reside? Is it in the agent or in the environment?

Alternatively, a third path is available which takes the view that the entire virtual ecology

[8]Developing more sophisticated reasoning algorithms and models has been the focus of significant research in both the artificial intelligence and multi-agent systems fields.

[9]This is analogous to the annotation and tagging of information on the World Wide Web, through technologies such as XML. This allows web pages or information exchanged via web services to be meaningful, accessible and easily processed by web based information agents.

(agent and environment) needs to be viewed as a whole and that not only will agent and environment need to change, but more importantly, the interface between the two needs to go beyond just the exchange of percepts and actions.

Reconsidering the agent-environment interaction involves a number of different aspects. First, it involves selecting appropriate and agent suitable representations of the virtual environment. This could range from adding simple agent readable textual annotations to the virtual world, or it could be more significant and involve a major rethink as to how virtual environments should be structured to make them amenable to agent systems.

Second, it involves deciding on the details of the agent-environment interaction. What is it that the agent can actually perceive and what actions can it take in the world? How are the percepts and actions related and constrained by the agent and the environment? Ideally, agents should be able to go beyond the perception of simple labels and annotations in the environment. Each agent should be able to perceive annotations which are not only tailored to it, but are also context sensitive, purposeful and action oriented in nature.

Given that one of the drivers for this research is to better model and represent military operators in military simulations, it makes senses to look at ideas from the cognitive sciences for insight. Ideas such as affordance theory from ecological psychology can inform the multi-agent simulation designer to better capture, model and represent the agent-environment interaction.

Affordance theory describes how humans and animals can perceive opportunities for action (or action possibilities) with respect to the entities in the world around them. These opportunities for action are referred to as affordances, because a particular object may afford a human or an animal a particular course of action. Affordance theory provides a suitable starting point for understanding and exploring the agent-environment interaction. The agents (in this case humans or animals) go beyond the ability to perceive simple labels and annotations, and can perceive action possibilities that are relevant, tailored and context sensitive to them. Furthermore, affordance theory provides a mechanism that directly relates perception to action.

1.3 Military Multi-Agent Simulation

Modelling and simulation [28] have a long history of being used to support, analyse and understand military operations. Modern operations research had its genesis in the Second World War where science and mathematics were used to understand complex problems and to ultimately provide advice to military decision makers about military operations and weapons [116, 9]. In recent times, multi-agent simulation is but one tool which operations researchers have used to model and simulate military operations [114].

Computational simulation in operations research is typically used to provide advice for a number of different purposes. These include providing advice on the acquisition of new aircraft and systems and the development of tactics and concepts of operations [10] for equipment currently in use.

[10]Concepts of operations (or CONOPS) refers to how a particular military capability will be used and operated within a particular operational context.

In both the acquisition of military hardware and in the development of tactics it is critical that the analysis produced by operations research techniques and the resulting advice provided is credible, reliable, verified, validated and explainable. Hence there are specific constraints and requirements that are placed on the development of multi-agent simulations used for these purposes.

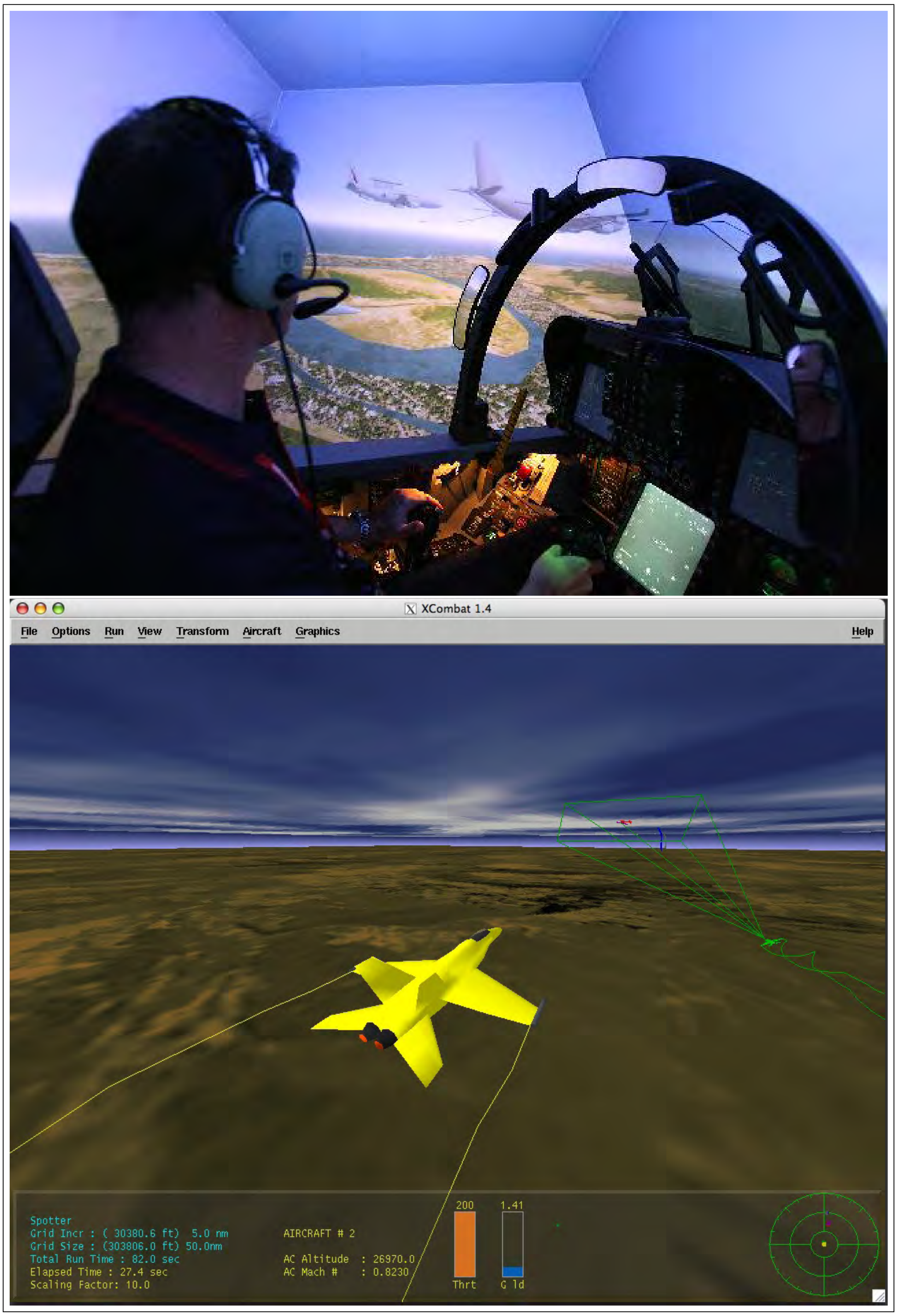
Military simulations typically consist of computational representations of a battlespace. For example, in an air combat simulation this would include computational models of aircraft dynamics, sensors, communications systems, weapons as well as representation of the physical environment such as the terrain and atmospheric conditions in which the scenario is being played out. Intelligent agents are typically used in these types of simulations to represent the reasoning and decision making processes of military operators such as fighter pilots [70].

One of the earliest applications of intelligent agents has been in their use as computational models of fighter pilot reasoning in military simulations of air combat [70, 148, 151]. A number of different agent technologies have been applied to modelling air combat in multi-agent simulations. Perhaps the best known are the use of SOAR [130] in the tactical air combat simulation known as TacAir SOAR [79], and the use of agent programming languages based on the BDI agent reasoning model [121] such as dMARS [33] and JACK [74].

This research was motivated by the application of intelligent agents in the simulation of air combat. It is part of a larger research program that originally started out looking out how to better model fighter pilot situational awareness in multi-agent simulations used for air combat operations research and has included work in:

- Using BDI based agent programming languages to simulate complex air to air combat engagements [94, 92, 95].
- Methodologies for interchanging agents and humans in military simulations [62].
- Modelling teams of agents in complex organisational structures such as in a military command and control structure [154, 57, 155, 152].
- Looking at specific instances of agent-agent interaction such as modelling one agent being able to recognise the intention of another agent [67, 66, 64, 150, 122].
- Investigating the use of software engineering tools and methodologies to help manage the engineering of complex multi-agent simulations [112, 56, 65].
- Investigating techniques for computational cognitive modelling [100, 63].
- Approaches and techniques for the intelligent application of agent technology to modelling and simulating military operations [70, 68, 151, 71]

While there are many types of military simulations used for different purposes, in general one can think of two general categories. These are human-in-the-loop simulations and constructive simulations, examples of which are shown in Figure 1.4. Military multi-agent simulations typically can be categorised as those that include human operators and those that are completely constructive in which all human roles are filled by agents.



```
Spotter
Grid Incr : ( 30380.6 ft)  5.0 nm
Grid Size : (303806.0 ft) 50.0nm
Total Run Time : 82.0 sec
Elapsed Time : 27.4 sec
Scaling Factor: 10.0
```

```
AIRCRAFT # 2

AC Altitude  : 26970.0
AC Mach #    : 0.8230
```

Figures 1.4: Examples of two different air combat simulations. (Above) A human-in-the-loop training simulator and (Below) an air combat constructive simulator where the fighter aircraft are flown by intelligent agents.

Human-In-The-Loop Multi-Agent Simulation: Human-in-the-loop simulations typically have one or more human operators filling the role of a pilot or other military operator. These types of simulations often have immersive displays and human interfaces allowing a human to operate an aircraft in the simulated world. They are often used for both training purposes and as a mechanism for evaluating tactical procedures and new concepts of operations before they are fielded in real world exercises. In both cases intelligent agents can be used as team mates or adversaries. In this context, these computational representations of military operators are often referred to as Computer Generated Forces or CGFs [93].

Constructive Multi-Agent Simulation: Constructive simulations do not involve human input at run time and can be run in batch mode to allow operations analysts to conduct statistical studies comparing the effectiveness of different systems or tactics in different scenarios. [11] Instead of human operators directly participating in the simulation, the decisions of a military operator such as a fighter pilot are typically represented using intelligent agents.

It is important to note that constructive and human in the loop simulators are by no means mutually exclusive. In many cases constructive simulations used to conduct operations research studies have been modified to allow for one or more of the intelligent agents to be replaced by a human operator performing a particular role in real time. Similarly, intelligent agents can be used as virtual adversaries or team mates in human-in-the-loop simulators which are used in a training capacity.

In the case of simulations used for training, the inclusion of the human element means that both the aircraft systems controls as well as the graphical representation of the world have to be sufficiently detailed to provide realistic, immersive and ultimately useful training outcomes for the pilot or other aircrew. The need to provide realistic and immersive visuals means that the design of the virtual environment is heavily driven by the needs to efficiently render a realistic scene for the aircrew undergoing training.

Integrating agents as team mates or adversaries in simulations which have been primarily designed with providing realistic and immersive visuals to humans can be difficult [153]. Typically the virtual environment is represented by a scene graph or other similar database which has been specifically designed for graphically rendering the world for a human participant. This does not necessarily make it suitable, accessible or amenable to intelligent agents situated in this environment.

One of the reasons that this is the case is that information in scene graphs is usually a representation of the physical visual world. However, agents are not only interested in visible physical entities in the world, but also in what possible actions they can take, the relationships between entities, and other aspects of the environment which are not visible in a graphical rendering of the physical world such as social and organisational structures.

Ideally, a virtual environment such as this should be designed up front to serve the needs of both humans and agents making use of it. That is the virtual environment should be

[11] Constructive simulations used for the purposes of operations research are typically run in a Monte-Carlo batch mode, producing statistical outcomes for particular measures of effectiveness. They typically also run at many times faster than real-time, allowing analysts to explore hundreds and thousands of variations in a study.

represented in a manner that is suitable and accessible to human and agents.

This is largely a question of environmental representation and design. If the key roles in a military simulation (such as that of a fighter pilot) are to be filled by either a human or a computational agent, then the virtual environment needs to be designed and represented in a way which allows for both humans and agents to have access to the same information in the virtual world so that they can be situated on an equal footing. This thesis proposes that the theory of affordances can be used as a stepping stone to achieve this goal.

The theory of affordances as applied to multi-agent simulations has the potential to make both existing and new virtual environments accessible and amenable to intelligent agents. This is because it allows for the environment to be viewed by an agent in a context sensitive and tailored manner and it can also take into account a number of different and abstract concepts of the environment, going beyond what is physically visible.

Affordances are context sensitive, relevant and meaningful action possibilities for agents. Constructing a virtual environment which allows for the perception of affordances by agents means that the environment is more meaningful, accessible and amenable to agents.

Affordances also introduce a common language for both humans and agents to described how they perceive the world. The ability to explain and understand behaviour whether it be of a human or an agent in a simulated scenario is important in operations research.

Explanation leads to insight that helps to develop new concepts of operations. Affordances help to explain why a human undertook a particular action in the world. Most importantly intelligent agents could perceive and adopt simulated affordances in a virtual environment, making it possible to talk about the behaviour of both humans and agents participating in a simulation in terms of the courses of actions that were afforded them.

The idea that a modelling choice can make the observed behaviour of a model easier to explain and understand is not new. In the domain of multi-agent simulation of air combat, two particular modelling choices for agent reasoning have had a significant impact on the ability of analysts, subject matter experts and software engineers to understand and explain simulated tactical behaviour [71].

1.4 Affordance Based Interaction

The approach to affordance based agent – environment interaction adopts a number of stances primarily from the situated cognition and ecological psychology communities.

Situated Agency The principle that intelligent behaviour is the result of the interaction between an agent and the environment it is situated in, rather than being the result of purely agent reasoning.

Direct Perception The principle from ecological psychology that agents can directly perceive information in the environment. That is, meaningful information is available in the environment for agents to perceive without undertaking sophisticated reasoning about what they are perceiving.

Perception of Affordances The principle that agents can directly perceive opportunities for actions or action possibilities in the world that *afford* them a specific course of action. Furthermore, the principle that these affordances can be directly perceived in the environment.

Explicit Environmental Representation The principle that representations of a virtual environment in a multi-agent simulation needs to be explicit, and that for many problems pure multi-agent systems which involve a set of interacting agents do not sufficiently capture agent-environment interaction.

An affordance based approach for modelling the agent-environment interaction was selected for a number of reasons. Affordances allow agents to perceive the environment in a manner that is meaningful to them. That is, the affordances an agent perceives will be influenced by an agent's actual and relative position in the environment, as well as by what the agent is trying to achieve in the world. The affordances an agent perceives will depend on the agent's goals, making affordances intentional in nature. Affordances also closely relate an agent's percepts to the possible actions that they can undertake in the environment.

Developing credible computational models of human decision making and behaviour for use in applications such as military simulation requires at least some foundation in accepted theories of cognition. How strongly a computational model adheres to these theories will depend on the application domain and the nature of the problem being solved. This is in contrast to some application domains such as in interactive entertainment (e.g. video games). In most video games it is not important if the theory and design behind the artificial intelligence in a character agent is based on a scientific theory. Rather, the emphasis is on providing an illusion of intelligent behaviour for the purpose of entertaining the game player.

1.4.1 Managing Simulation Complexity

An affordance based approach to modelling the interaction between agents and virtual environments has the potential to simplify the complexity of multi-agent simulations. Complexity in these types of systems can take on a number of different forms.

These include the complexity of the overall simulation architecture, the complexity of individual subsystems (such as those of the agents or the environment) in the simulation, as well as the complexity inherent in the interaction between these subsystems.

Like many other types of software systems, the implementations of military simulations have over time transitioned from imperative programming languages and approaches, to object-oriented languages and methodologies. With this transition came advances in software engineering tools, techniques, methodologies and programming languages that helped manage the complexity of the development of large software systems. Although less mature and widespread, there have been similar advances in the development of agent-oriented programming languages, tools, techniques and methodologies.

Many military multi-agent simulations are designed using hybrid agent-object oriented software architectures. This architectural split represents the different approaches taken to modelling the various components of the simulation.

For example, in some of these simulations computational cognitive models of military decision makers are implemented using agent programming languages. On the other hand, models representing physical systems (such as the aircraft, sensors, weapons and the rest of the environment) are implemented using object oriented programming languages.

If each component of the system is treated as separate, then tools, techniques and methodologies are available and in use to help manage, design and implement both the agent and object oriented components.

Understanding and modelling the interaction between agents and their environments in a multi-agent simulation has not been seriously addressed. This becomes a significant software complexity management issue especially due to the requirement for representing increasingly complex tactical behaviour in many military simulations. This increase in behavioural complexity means that the agent – environment interaction becomes more complex and hence there is a need to appropriately manage, model, design and understand it.

From a software design perspective the complexity arising from more sophisticated agent behaviour can be addressed in one of three ways; in the agent, the environment or the interaction between the two as summarised in Figure 1.5.

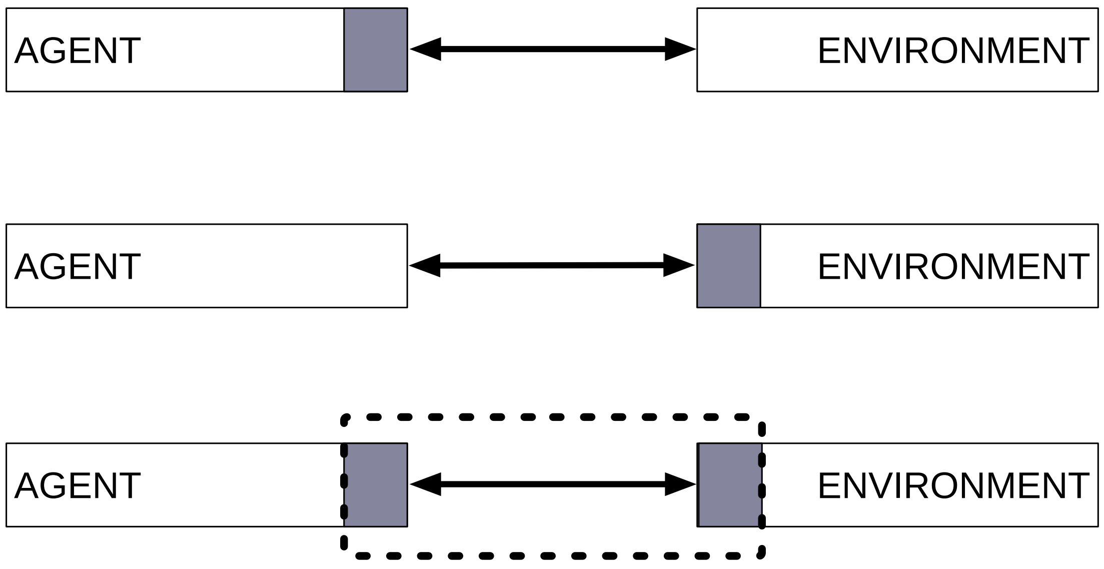
Agent It can be addressed in the agent by designing and building a more sophisticated agent. An increase in the agent complexity manifests itself in both the perception and reasoning components of the agent architecture resulting in increased development time, reduced computational performance as well as making it more difficult for the encoding of complex behaviour to be understood.

Environment Alternatively, complex intelligent behaviour can be developed by making the virtual environment more intelligent. This may take the form of appropriately labelling and annotating the virtual environment with information that is useful and meaningful to the agent. This has the potential to simplify the design of the agent and increase computational performance at the expense of making the virtual environment more complex.

Agent-Environment Interaction A third approach is to consider the entire agent-environment interaction. That is, the design of complex and intelligent behaviour in multi-agent simulations should take into account the design of the agent, the design of the environment and most importantly the design of the interaction between the two.

Rather than designing this interaction on an ad-hoc basis, the theory of affordances provides a basis for a model of interaction that can help to manage this complexity. A model of agent-environment interaction inspired by the theory of affordances can take into account aspects of the agent, the environment and their relationship in a well defined and consistent fashion.

Affordances are essentially about the interaction between humans (or animals) and their environment. Affordances provide ecological psychologists with a model of this complex interaction. It is conceivable that a similar model can be applied to understanding and managing the complex interaction between computational agents and virtual environments.



Figures 1.5: The need for more sophisticated intelligent behaviour can be addressed in one three main ways. (Top) Designing a more complex and intelligent agent. (Middle) Designing a more complex and intelligent environment. (Bottom) Designing a more complex and intelligent interaction between agent and environment, which is the approach adopted in this thesis. The theory of affordances is used as a mechanism for modelling and understanding this interaction.

1.4.2 Computational Performance of Simulations

Taking an affordance based approach to representing the agent-environment interaction does not necessarily suggest an impact (either positive or negative) on the computational performance of a multi-agent simulation. However the corresponding design and implementation decisions will have an impact on the computational performance of the simulation.

The introduction of an affordance based model of interaction means a higher level model of fidelity for the agent, the environment and their interfaces. A higher level of model fidelity does not necessarily mean additional complexity, but does have the potential to significantly affect simulation computational performance.

By taking innovative approaches to the design of these simulation systems, the introduction of affordances has the potential to reduce the computational impact on some components of the simulation system. For example, a design where affordances are computed in the environment and in the interaction between agent and environment has the potential to reduce the computational load on an agent component.

1.4.3 Authentic and Cognitively Plausible Agent Behaviour

Affordance theory also has the potential for higher fidelity models of agent-environment interaction in multi-agent simulation. A design based on a widely accepted theory of interaction from the cognitive sciences also means that the model of interaction in these multi-agent simulations are more cognitively plausible, more authentic and more credible in their representation of how humans interact with their environments. This is particularly important in multi-agent simulations of real world systems such as in military operations research.

The computational models that represent aircraft and other entities in military simulation have a basis in well known and understood scientific and engineering theories and models. For example, the design of sensor, aircraft and weapons models are based on engineering specifications of these systems as well as dynamics and electromagnetic propagation models from physics. Similarly models of military decision making implemented as intelligent agents tend to be based on models from cognitive psychology. This is critical for producing credible and plausible analytical results from these simulation systems.

The tactical behaviour exhibited by these agents results from the interaction between the physical and cognitive models. Ecological psychology and in particular affordance theory provide a suitable model to design a framework on which this interaction can occur in the virtual battlespace. Given the importance of the resulting tactical behaviour in many military simulations, which are highly dependent on the agent-environment interaction, it makes sense to select a model of interaction that is not only of a sufficiently high fidelity but also one that is cognitively plausible.

Humans perceive meaning in their environment through the affordances that they perceive. Modelling affordances in a multi-agent simulation means that the agents inhabit virtual worlds which are meaningful to them. This not only lends to the cognitive plausibility of a simulation but also means that the virtual environment is more amenable to easily integrating agents.

One of the advantages of using intelligent agent technology is the ability to explain, understand, verify and validate tactical behaviour more clearly. This is particularly the case with agent languages based on the BDI model, which make use of programming constructs that correspond to mental attitudes such as beliefs, desires and intentions; providing a link between agent programmers and subject matter experts.

For example, in an air combat simulation the ability to explain tactical behaviour using the constructs provided by the agent programming language means that the verification and validation process becomes more accessible to subject matter experts.

In a similar manner, affordance based models of agent-environment interaction have the potential to play a similar role in making it easier to verify and validate tactical behaviour in multi-agent simulations.

Affordances can be used to describe the specific behaviour undertaken by humans in their environment in terms of the action possibilities available to them. Multi-agent simulations which make use of affordances as a mechanism for modelling the agent interaction with the environment allow for agent behaviour to be explained in a similar fashion.

The ability to explain behaviour using terminology and language which can be understood by a domain expert becomes an invaluable tool in the process of verifying and validating these simulations.

1.5 Contributions

This thesis makes two primary contributions. These are a model of agent – environment interaction for multi-agent simulation inspired by theory of affordances from ecological psychology, and a set of lessons learnt and insights gained on implementing an affordance based approach to agent – environment interaction in practical multi-agent simulations. Specifically the contributions are:

(1) Affordance Based Interaction Model A formal model of agent – environment interaction inspired by the theory of affordances is presented in Chapter 5. There are a number of characteristics which make this model a novel and original contribution. It incorporates aspects of affordance theory which are relevant and important to multi-agent simulation. The model goes beyond representing affordances as simple static annotations in the virtual environment. Rather it represents affordances as action possibilities that are dynamic, relational, intentional and directly perceivable. Furthermore the model is framed in the context of Boyd's OODA (Observe, Orient, Decide, Act) model of military decision making [29]. Importantly the work demonstrates how to incorporate relevant aspects from the beliefs, desires and intentions (BDI) model of agent reasoning [120, 121].

(2) Analysis of Impact of Affordance Based Approach In Chapter 7 the lessons learnt and insights gained from the development of the affordance oriented model and its implementation in two practical multi-agent simulations described in Chapter 2 and 6 are presented. The contribution is in the form of an analysis of the impact on a multi-agent simulation if an affordance based approach is adopted. The impact on a multi-agent simulation is addressed in the form of three different areas. (a) The impact on the design of multi-agent simulations. (b) The impact on the computational performance of multi-agent simulations. (c) The impact on the application domain making use of affordance based multi-agent simulations.

Two different practical multi-agent simulation implementations incorporating the affordance model are described in Chapters 2 and 6.

The first system developed was a complex, real world, military multi-agent simulation called the Human Agent Virtual Environment (HAVE) and is described in Chapter 2. In this simulation the role of a strike fighter pilot undertaking a Close Air Support (CAS) mission [21] could be undertaken either by a human pilot or a computational agent pilot in the single common virtual environment. The pilot agent in this simulation implements reasons and makes decisions about the entities in the virtual world in terms of the affordances perceived.

While this system makes use of the model and the ideas presented in later chapters, the description is presented early on in the thesis because it acts as a motivational and aspirational example of the type of complex application domain in which an affordance based approach can make difference in the design of the simulation.

The second system was a multi-agent simulation of the game Capture The Flag (CTF) [6] and is described in Chapter 6. This simulation interpreted the affordance based interaction model as two different designs; one where affordances are computed in the environment and are perceivable by the agents in the game, and the other where each agent computes the affordances that are relevant to it.

The system was developed to allow for an empirical evaluation of the affordance based model of interaction in a simulation environment that did not come with the complexities of a full blown and complex domain such as the HAVE system, yet still maintained characteristics that were typical of a military simulation. That is, it was still desirable that the simulation was adversarial, dynamic and continuous in nature yet was still simple enough to conduct a meaningful empirical analysis comparing variations of the affordance based model.

The most important insight gained from the development of these two multi-agent systems was that the affordance model of interaction described in Chapter 5 was a viable approach to agent reasoning. Furthermore, the implementations helped to evaluate the impact an affordance based approach has on a multi-agent simulation, from the perspective of design, computational performance and the effect it has on the ultimate use of the simulation in the specific application domain in which it is being used.

The development of the model and subsequent implementation in the two multi-agent simulations led to the identification of number factors important to affordance based multi-agent simulations. These include:

- An explicit representation of the virtual environment is critical for any introduction of an affordance based model of interaction in a multi-agent simulation.
- The affordances available to humans in the real world are not only dependant on the physical environment but are often also influenced by other environmental modalities such as the social environment. To better capture the concept of an affordance in a multi-agent simulation, representation of the virtual world should be multi-modal. That is, they should not just represent the physical environment but also represent other environmental modalities that are relevant to the specific application domain.
- There are a number of ways of implementing an affordance based approach in a multi-agent simulation. Which approach a designer of a multi-agent system adopts will depend on design, performance and model fidelity trade offs. The two interpretations looked at in this work involved (a) an intelligent agent approach where an agent determines the affordances available to it and (b) an intelligent environment approach where the interaction between agent and environment is considered by the virtual environment.
- An intelligent agent approach to determining affordances means that affordance based reasoning can be introduced to an existing multi-agent simulation by incorporating the model inside the agent. The alternative approach, while having a greater impact on the design of the multi-agent simulation is also more consistent with the view of affordances from ecological psychology, where affordances are considered directly perceivable action possibilities in an environment.

- Affordances can be considered as a special type of annotation in a virtual environment. In their simplest form these annotations are just simple labels which describe an action an agent can take with respect to an entity in the world. However, this work shows that these annotations can be more sophisticated – not only being dynamic, but being context sensitive and tailored to each individual agent.

A number of insights were gained from the set of experimental results comparing the computational performance of different design interpretations of the affordance model in the Capture The Flag simulation. In the context of the CTF simulation, a number of important conclusions can be drawn from these results.

- The subsystem responsible for computing affordances in the simulation system can have a potentially significant computational footprint compared to other subsystems. This is especially the case if affordances are computed dynamically (at run time) as opposed to being pre-computed.
- The overall computation time of the simulation is comparable whether affordances are computed in the agent or in the environment.
- A general approach to computing affordances can have a significant impact on computational performance. By taking domain specific information into account, the algorithm tasked with evaluating the affordances in a simulation can be made more efficient significantly increasing the computational performance of the simulation.

1.6 Thesis Structure

This thesis can be divided into a number of parts spread over a total of eight Chapters. The main contribution of the thesis, a model of agent – environment interaction motivated by affordance theory is described in Chapter 5. A high level background overview of some of the issues associated with taking an affordance oriented approach to these types of interactions is presented in Chapter 4.

Chapter 1. Introduction This Chapter has described the approach taken to modelling agent – environment interaction in multi-agent simulations in this thesis as one based on theory of affordances. The Chapter describes existing approaches to agent – environment interaction followed by a description of research in military multi-agent simulation which directly motivated the research in this thesis. This is followed by a description of the affordance based interaction approach adopted including an overview of what this means for a number of different areas of multi-agent simulation. The contributions made by this research are then described.

Chapter 2. Motivation: The Human Agent Virtual Environment This Chapter describes a complex multi-agent simulation in the military domain of Close Air Support (CAS). While the implementation of HAVE is based on core work presented later in the thesis (in particular Chapters 4 and 5), it serves the purpose of motivating the work and providing concrete examples of the kinds of affordances important in the domain. HAVE represents not only a motivating example of a viable approach to incorporating affordances into a complex military multi-agent simulation but also an aspirational example. This is because the HAVE implementation allows for the practical evaluation of the model in a complex and real world application domain and because it represents a virtual environment in which humans and agents can interact with each other.

Chapter 3. Literature Review: Situating Agents in Virtual Environments This Chapter reviews existing techniques for situating agents in intelligent environments. Specifically it looks at work that has been done related to using intelligent agents as cognitive models to simulate human behaviour particularly in the domain of military simulation. It describes the Beliefs, Desires and Intentions model of agent reasoning as well as Boyd's OODA loop model of military decision-making, in the context of military multi-agent simulation. The Chapter also provides an overview of related work in which affordance based approaches have been used in multi-agent simulation.

Chapter 4. Background: Affordance Oriented Interaction This Chapter introduces the background constructs for modelling affordances and discussed the implications for the design of the agent-environment interaction, the design of the environment and the design of the agent when implementing an affordance based approach.

Chapter 5. Modelling Affordances: Reasoning About Agent – Environment Relations This Chapter presents the new model of agent – environment interaction inspired by the theory of affordances. The model is presented both in a descriptive form as well as a formal mathematical model. This Chapter includes a review of the literature from ecological psychology on the nature of affordances that is relevant to the formulation of the model. A number of illustrative examples are included from the game Capture The Flag (CTF).

Chapter 6. Evaluation: Capture The Flag This Chapter presents the results of an empirical, comparative evaluation. It investigates the impact of different design decisions associated with implementing affordance based multi-agent simulation architectures. A multi-agent simulation system is constructed, implementing the game of Capture The Flag. The single simulation architecture is designed to allow multiple approaches to implementing affordances based agent-environment interaction. A comparison and evaluation of the multiple approaches is undertaken including conducting a series of experiments that assess the impact of different designs on computational performance.

Chapter 7. Discussion This Chapter brings together the insights gained from the theoretical development of the affordance models and the two different implementation architectures.

Chapter 8. Conclusions This Chapter provides a discussion and summary of the research including an evaluation of potential application domains. The major contributions and limitations of the work are described as well outlining the areas and directions where this research could be extended.

Chapter 2

Motivation: The Human Agent Virtual Environment

"Without an environment, an agent is effectively useless. Cut off from the rest of its world, the agent can neither sense nor act. An environment provides the conditions which an entity (agent or object) can exist. It defines the properties of the world in which the agent will function. Designing effective agents requires careful considerations of the physical and communicational aspects of their environment"

— James Odell et. al [106]

2.1 Introduction

This Chapter introduces a multi-agent simulation implementation that motivates the work in this thesis. The simulations was developed as a way of exploring the ideas on modelling affordances and multi-modal virtual environments in the context of a complex military multi-agent simulation.

The resulting implementation of an affordance model in a real world, complex multi-agent simulation, is named the Human Agent Virtual Environment [113] or HAVE.

The primary research driver in the development of HAVE was to explore representations of virtual environments in which both humans and agents are situated, perceive these environments and undertake meaningful and appropriate actions.

Three important research challenges have been addressed by the work. The first, is the implementation of a multi-modal representation of the virtual environment — having multiple, parallel yet consistent representations of the virtual world that were accessible to, and tailored for the different simulation components. The second is the use of labelled annotations in the virtual world which the agents could easily access and interpret. The third, the use of an appropriate architecture for capturing and representing interaction between agents and the environment they are situated in.

HAVE models and simulates a Close Air Support (CAS) mission which involves fighter or strike aircraft providing support to ground troops through the use of air-to-ground weapons.

This provides a realistic and relevant domain in which to explore agent-environment interactions. An example screen capture from HAVE is shown in Figure 2.1.

HAVE was specifically designed as a virtual environment in which agents and humans could interact. Consequently although it shares many design choices and characteristics that are present in both constructive operations research military simulators and human-in-the-loop training simulators it does posses some unique properties.

The development of HAVE was motivated by a desire to better understand agent interaction with a complex virtual environment, with affordance theory being one of many possible ways to model this interaction. The development of this simulation was motivated by a number of factors.

- To explore the viability of an affordance based approach to agent reasoning in a real world application domain set in a rich and complex virtual environment.
- To take the concept of a multi-modal virtual environment one step further by incorporating representations of the virtual environment that is accessible to humans (via graphical rendering of the virtual world) while at the same time keeping them consistent with the representations of the virtual world that have been designed for agents.
- To look at the concept of affordances as a unifying concept for allowing humans and agents to interact in a common virtual space.

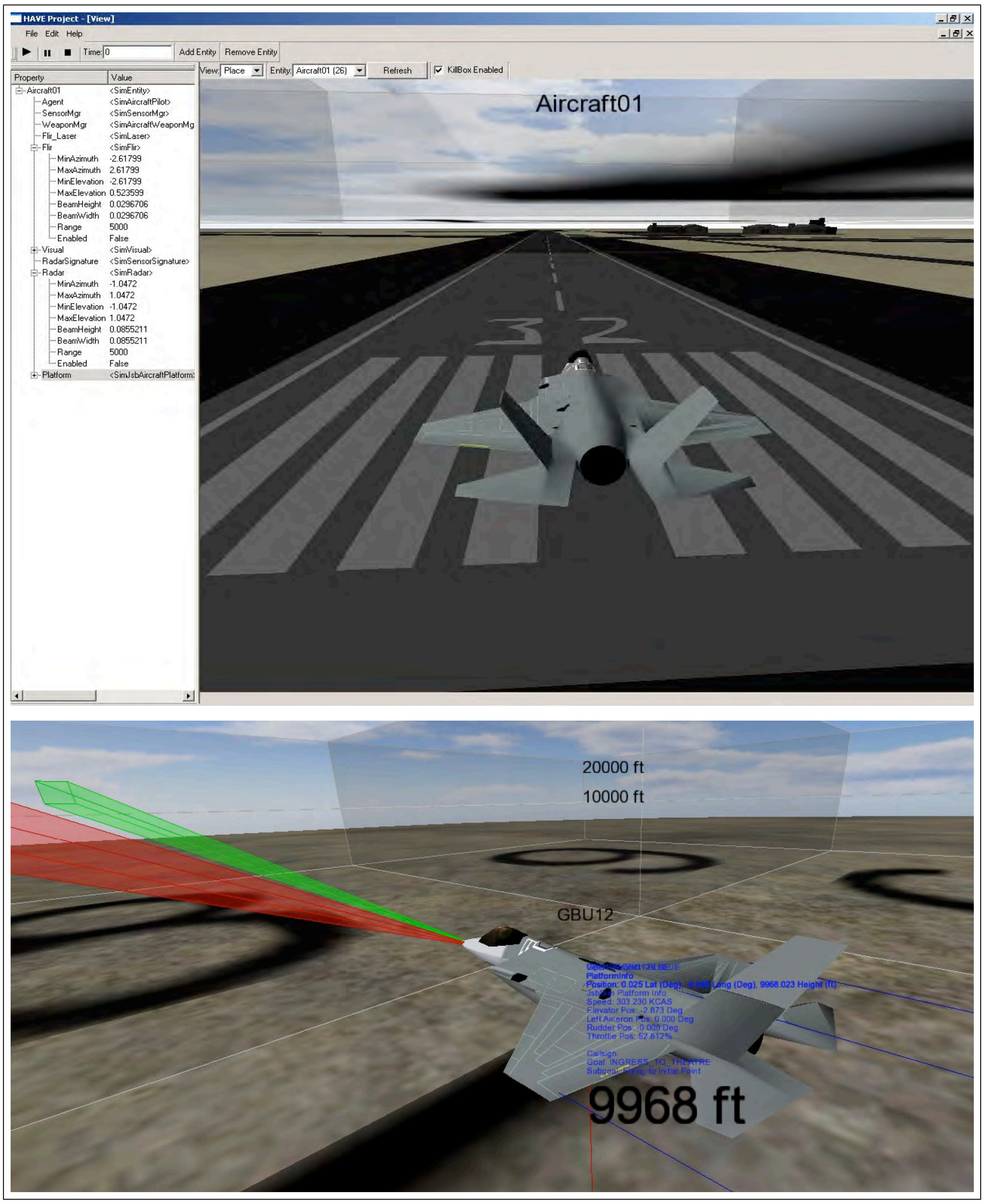
The Chapter begins with a description of why issues such as human-agent interaction are important in military simulations. This is followed by a brief explanation of the domain of Close Air Support.

A significant part of the Chapter deals with describing the design of the HAVE architecture with specific emphasis on the representation of the virtual environment. This is followed by an account of how the strike fighter pilot is modelled in HAVE both from the perspective of the human pilot and the affordance based agent pilot. Finally the Chapter describes the primary lessons learnt from the development of HAVE.

2.2 Human-Agent Interaction in Military Simulation

The inclusion of human participants in military simulations has become increasingly important in many different areas from operations research, to training simulators, to military experimentation [157].

A virtual environment which allows for the role of a military operator such as fighter pilot to be undertaken by either a human or an agent has many benefits. The most obvious benefit is that agents can undertake the role of virtual opponents or virtual team mates in a study, training exercise or experiment.



Property	Value
Aircraft01	<SimEntity>
Agent	<SimAircraftPilot>
SensorMgr	<SimSensorMgr>
WeaponMgr	<SimAircraftWeaponMg
Flir_Laser	<SimLaser>
Flir	<SimFlir>
MinAzimuth	-2.61799
MaxAzimuth	2.61799
MinElevation	-2.61799
MaxElevation	0.523599
BeamHeight	0.0296706
BeamWidth	0.0296706
Range	5000
Enabled	False
Visual	<SimVisual>
RadarSignature	<SimSensorSignature>
Radar	<SimRadar>
MinAzimuth	-1.0472
MaxAzimuth	1.0472
MinElevation	-1.0472
MaxElevation	1.0472
BeamHeight	0.0855211
BeamWidth	0.0855211
Range	5000
Enabled	False
Platform	<SimJsbAircraftPlatform:

```
[illegible]
PlatformInfo
Position: 0.025 Lat (Deg), [illegible] Long (Deg), 9968.023 Height (ft)
JsbSim Platform Info
Speed: 303.230 KCAS
Elevator Pos: -2.873 Deg
Left Airleron Pos: 0.000 Deg
Rudder Pos: -0.000 Deg
Throttle Pos: 52.812%

Callsign:
Goal: INGRESS_TO_THEATRE
Subgoal: Flying to Initial Point
```

Figures 2.1: Visualisation of a strike fighter aircraft taking off from an airport (above) and in the air (below) in the Human Agent Virtual Environment (HAVE). The coloured sectors emanating from the nose of the aircraft in the second image denote the sensor coverage of the strike fighter aircraft. In this particular case the coverage of the radar is denoted by the red volume and the coverage of the infra-red sensor. In both cases the sensors are in a narrow beam mode.

However constructing virtual environments capable of supporting hybrid human – agent teams poses many different challenges. The challenge most relevant to this research is related to how the virtual environment is represented in these types of simulations.

There is a disconnect between virtual environments which have been designed for human use [12] and hence are not necessarily easily accessible to computational agents, and the virtual environments designed specifically for computational agents.

An affordance based approach to designing these environments and their interaction with computational agents has the potential to provide a common framework and language for both humans and agents. That is, complex decisions made in the course of a simulation by either computational agent or human operators can be explained and understood using the common concepts of action possibilities and affordances.

This has the potential to be an extremely powerful tool in exploring complex tactical behaviour and concepts of operations; especially in the case of hybrid human – agent teams. The development of HAVE is a first step towards this longer term aspirational vision of human and agent interaction in common virtual environment.

Specifically, two areas are addressed. The first is that it incorporates an affordance based model of agent reasoning to provide a mechanism for explaining, understanding, verifying and validating intelligent behaviour using concepts that are applicable to both human and agent behaviour in the one common virtual environment.

The second is that HAVE provides a reference design architecture illustrating how a multi-modal virtual environment can be designed to accommodate both human and agent representations of military operators.

2.3 Simulating the Close Air Support Mission

> *"Close air support has been around since the early days of the aircraft in World War I. Although it has gone under other names, such as army cooperation, close cooperation, and ground support, every, major country with an air force has tried it in some form.*
> *...*
> *First, by his very nature, the soldier on the ground will find close air support useful in almost every conceivable situation, from pursuit to retreat. If it were available, the man on the ground would like to see air precede his every move."*
> — Col. John A. Warden III, USAF [169]

HAVE implements a simulation of a Close Air Support or CAS mission. Close Air Support is a type of air operation in which fighter or strike aircraft provide support to ground troops through the employment of airborne weapons. The Royal Australian Air Force Air Power Manual [21] defines close air support as follows.

[12] Virtual environments designed for human use refer to environments typically represented using technologies such as scene graphs which have been designed and optimised for the graphical rendering of the visual and physical environment. These types of virtual environments are quite common in applications such as training flight simulators.

Figures 2.2: (Above) A Royal Australian Air Force F/A-18 Hornet undergoing refuelling prior to a Close Air Support (CAS) mission. This aircraft was the primary strike fighter modelled in HAVE. (Below) Australian JTACs requesting close air support from Royal Australian Air Force F/A-18 Hornets.

Definition 2. *Close air support is air action against hostile targets which are in close proximity to friendly forces and which require detailed integration of each air mission with the fire and movement of those forces.*

Typically a specially trained military operator on the ground known as a Joint Terminal Attack Controller (JTAC) [13] will call in an air strike over radio and guide or *talk-on* a pilot onto a target that is posing a threat to ground troops.

The instructions to locate the target that the JTAC on the ground provides the pilot who is in the air must be concise and precise. The instructions may include a verbal description describing the target and its location relative to local landmarks or it may include precise coordinates, or even the designation of the target using a ground based laser illuminator by friendly ground forces. It is the pilot's responsibility to positively identify the target, ensure all the rules of engagement are followed, and to subsequently engage the target. In modern CAS operations the use of precision guided weapons (such as satellite or laser guided weapons) aims to increase accuracy while minimising collateral damage.

CAS operations by strike fighter aircraft are usually conducted in pairs, with the lead aircraft dropping munitions while the wingman looks out for ground and air threats. These threats along with the requirements of flying low to the ground, strict rules of engagement, the need to avoid fratricide and civilian casualties as well as time sensitivities all make the CAS mission one of the most challenging a military aviator has to face.

The CAS mission was the domain selected for HAVE for a number of reasons. First, it provides interesting and differing interactions with the environment both from the ground perspective (friendly and enemy ground troops, JTAC) as well as from the air.

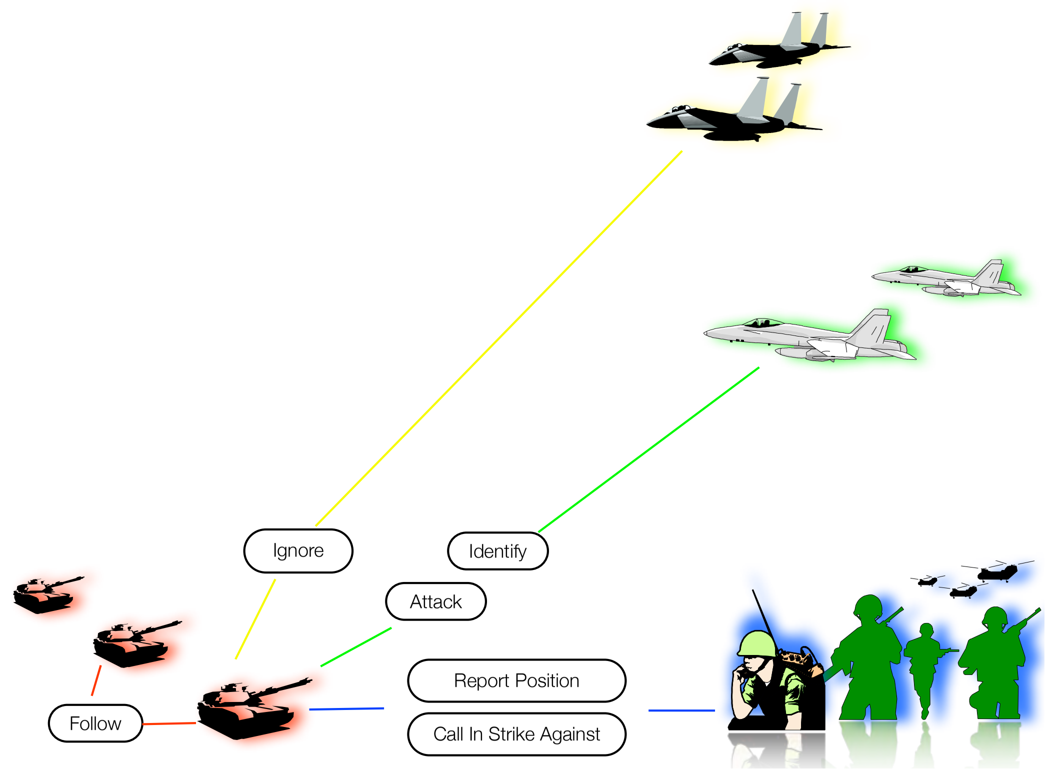
Second, it provides the opportunity for agents and humans to interact in the one virtual environment. A JTAC coordinating a strike on an enemy tank visible in the distance, with a pilot in a strike fighter seeing the same tank from 10,000 feet was a perfect example of a potentially interesting human-agent interaction.

Third, it allows for the interchanging of either a human or an agent to pilot the strike fighter aircraft and interact with the JTAC in the one virtual environment.

Fourth, understanding modelling and simulation of CAS missions was important and highly relevant in the context the types of operations undertaken by many air forces.

The combined air and land battle space in which military operations such as CAS missions are undertaken are extremely dynamic and dangerous environments. Military personnel must be able to continuously evaluate the possible actions available to them and make a decision as to which action to adopt under significant constraints. Affordances can be a useful way of thinking about a human's interaction with the world around them. This is particularly the case for strike fighter pilots during a CAS mission as understanding, evaluating and undertaking the actions available to the pilot from the environment can mean the difference between mission success and mission failure. Modelling the action

[13] Military personnel undertaking the JTAC role are known by a number of different names depending on the exact nature of the role, the specific service arm and the specific nation. The title Forward Air Controller (FAC) is also commonly used. It is also possible to control and guide strike aircraft from the air. In this case the role is typically known as an airborne forward air controller of FAC-A.



Figures 2.3: Affordances in a Close Air Support (CAS) environment. Different military operators perceive different affordances with respect to the same tank, depending on their side, role and the current situation.

possibilities perceived by a pilot during a mission can help to improve the understanding operations analysts have of how pilots interact with their environment.

Consider the affordances different people perceive with respect to a tank leading a convoy of tanks in a battlefield as illustrated in Figure 2.3. For the tanks in the convoy, the lead tank may afford following or may even afford protection. To ground troops on an opposing side, the tank is seen as a threat. The possible actions these ground troops can take with respect to the tank will depend on the exact role they are undertaking.

A JTAC equipped squad will have different affordances available to them. Possible actions such as reporting the position of the tank (or convoy) and calling in a strike become an option.

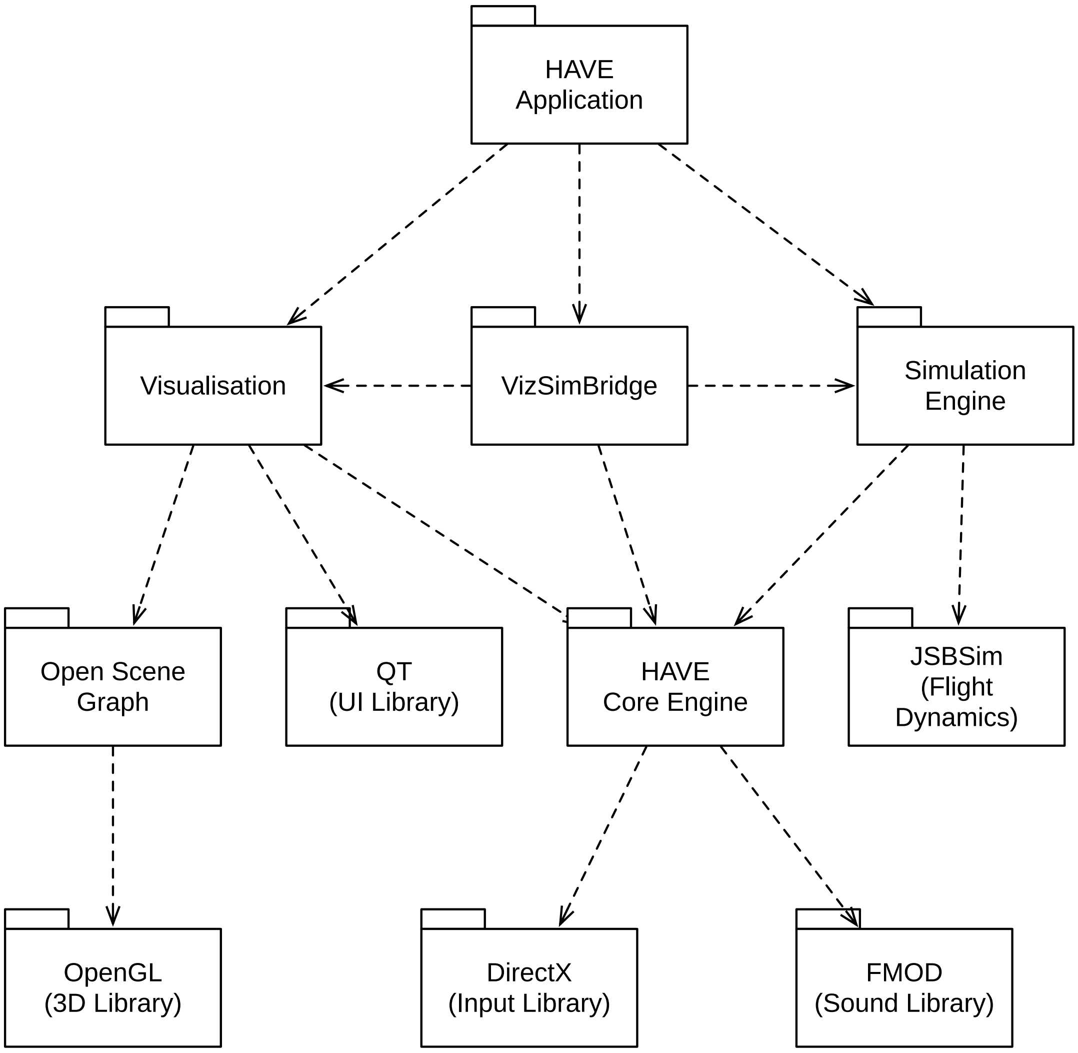
Pilots in aircraft flying above the battlefield will also perceive different affordances with respect to the tank convoy. For example, a pair of fighter aircraft flying a high combat air patrol (or CAP) [14] will likely ignore the tank. The tank affords ignoring because the role of the CAP is to patrol a region for enemy fighter aircraft rather than to engage ground targets even if the convoy is visible to them.

On the other hand a pair of strike aircraft tasked on a close air support or battlefield interdiction mission [15] will perceive very different affordances with respect to the tanks in the convoy.

For example, on first detection the tank does not afford being destroyed if it has not being identified as either a friendly or enemy tank. Therefore, the only action possibility available to the strike fighter pilot is to identify the tank. Once the tank has been identified as an enemy tank, this may not be enough to afford being destroyed. A strike against the tank may only be afforded under very specific situations. For example it might require a request for air support from a JTAC and authorisation to operate within a particular part of the battlefield.

Furthermore, the rules of engagement under which the strike fighter pilots are operating may prevent a strike if it placed civilians and civilian infrastructure at risk. If the strike fighters do not have any remaining weapons then the tanks also do not afford being destroyed. This is an example of where embodiment affects affordances.

As can be seen by this simple close air support example, just because an action (such as dropping a weapon) on a tank is possible, it doesn't necessarily make it an affordance. The affordances available to pilots and other war-fighters on the battlefield will be influenced by the pilot's intentions, role, embodiment, the environment, various relationships and by the current situation.



Figures 2.4: The major subsystems in HAVE, represented as a block diagram. The VizSimBridge subsystem act as a bridge between the visualisation and simulation subsystems ensuring the both virtual environments stayed synchronised. A number of well known standard technologies are used in the HAVE subsystems. The 3D visualisation was undertaken with OpenSceneGraph [19] and OpenGL. Parts of the DirectX library are used for human input devices such as a throttle and stick to fly the strike fighter aircraft and a steering wheel used to drive and control the tanks in the simulation. The graphical user interface (GUI) aspects of HAVE were built using the Qt library. Three dimensional audio effects are implemented using the FMOD sound library.

2.4 Overview of Software Architecture

The HAVE simulation consists of a number major subsystems. A high level overview of the major subsystems in HAVE is shown in Figure 2.4 represented as UML [16] packages. HAVE was developed implemented using the C++ programming language.

The HAVE Core Engine package provides a set of core utilities relevant to the air combat domain (such as navigational calculation utilities) that are used by both the simulation and visualisation engines.

The HAVE Application package brings all the various components together enabling HAVE to act as an application. It provides a graphical user interface to allow loading and saving of scenarios, editing of entity properties, changing views as well as processing user input from a variety of different input devices including mouse, keyboard, stick and throttle (to fly the strike fighter aircraft) and steering wheel and pedals (to drive the tanks).

The three most important packages in the HAVE architecture are the simulation engine, the visualisation package and the SimVizBridge, which links the two. These three subsystems are described in the following Sections.

Simulation Engine The simulation engine module consists of three important subsystems. These are the computational models of the various systems being represented in HAVE (such as aircraft, sensors, weapons and pilots), computational models of the environment (such as the physical and social environment), and the simulation kernel itself which executes all the models, and provides a mechanism for data exchange between the various components.

HAVE is a discrete time stepped simulation system [48] meaning that each model is executed at a regular time step simulating an interval of time. [17]

Visualisation The visualisation module provides the a view [18] into the three dimensional virtual environment from the perspective of a pilot sitting in the cockpit of a strike fighter aircraft.

This means it not only provides multiple three dimensional views of the world but also represents visualisation of the various instruments a pilot would have access to. This includes a heads up display (HUD) displaying information about the state of the aircraft as well as displays of the various sensor displays such as the radar and the forward looking infra-red (FLIR) sensor.

[14] A Combat Air Patrol or CAP is a fighter mission which patrols a region of sky with the aim of intercepting and engaging enemy aircraft.

[15] An interdiction mission attempts to strike enemy ground positions and infrastructure before they come into contact with friendly ground troops. Close air support missions however provide air support for ground elements which have come into contact with enemy ground troops.

[16] Unified Modeling Language.

[17] Although the default time step in HAVE is a tenth of a second, the time interval can be changed by the user at run time to change the grain of the simulation.

[18] The scene graph used to render the scene for the human pilot is constructed using Open Scene Graph (OSG) [19] which is built on top of OpenGL.

Viz-Sim Bridge This module is a bridge between the representation of the virtual environment in the simulation engine and the representation of the virtual environment presented to the human pilot using the visualisation module.

This module helps to maintain consistency between the different representations of the different virtual environments. This ensures that if there is a change in the virtual environment that is used by the agents in the simulation engine, it is reflected in the visualisation.

The *Visualisation* and *Simulation Engine* subsystems are two of the most important components of the architecture. The *Simulation Engine* not only consists of the simulation execution modules but also includes all modalities of the environment (including those suitable for intelligent agents).

2.5 Modelling the Environment

The design of the virtual environment in HAVE adopted a multi-modal approach to environmental representation, similar in some respects to the Multi-layered Multi Agent Systems (MMASS) model of Bandini et al [11, 10].

In addition to the benefits of an explicit representation of the environment, a multi-modal approach allowed for multiple representations of the virtual environment that were suitable for and accessible to the different systems being simulated. It also meant that the different aspects of the modern battlespace could be adequately captured and represented.

The various environmental modalities not only represented individual simulation entities but also explicitly modelled and represented the relations between these entities. The different modalities of the virtual environment represented in HAVE are shown in Figure 2.5.

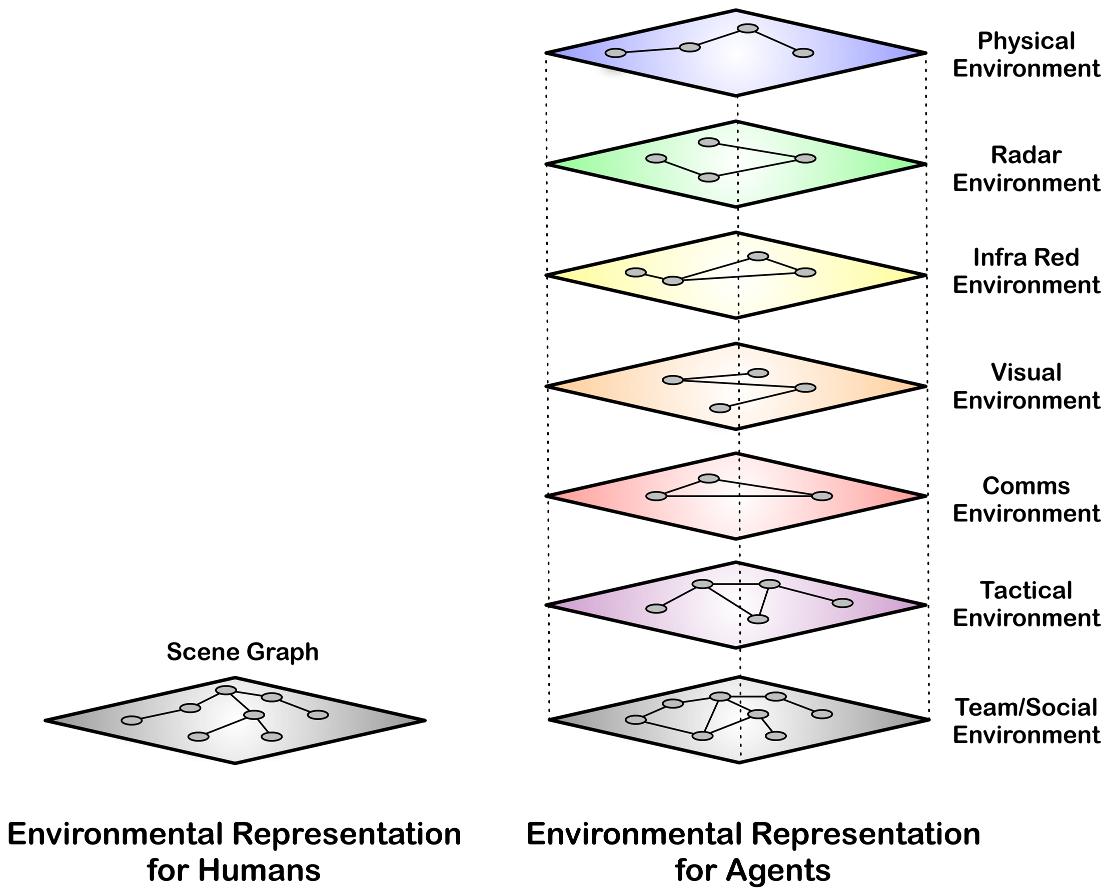
2.5.1 Design Challenges

The design of the virtual environment posed a number of challenges. The first challenge was to ensure that the different modalities captured the required information and relationships that were needed by the various systems being simulated. This ranged from representing aspects of the physical environment required for computational models of the flight dynamics of aircraft, to representing aspects of the electromagnetic environment required for various sensor models such as radar.

Additionally, aspects of the environment suitable for the intelligent agents in the simulation (in this case agents representing strike fighter pilots) also needed to be represented. This included annotating the relevant modalities with labels that were accessible and understandable by the agents in the simulation as well as representing the relevant military command and control relationships.

The second challenge was to ensure consistency between all the environmental modalities. For example this included ensuring that once an entity was destroyed that it was removed from all the relevant modalities.

The third challenge was to develop a representation of the virtual environment suitable for



Figures 2.5: Multi-Modal Environment in HAVE. HAVE represents two types virtual environment. A scene graph was used to represent the virtual environment in a form that was suitable for a human pilot. The rendering of the scene graph provided an immersive out the window view for the human pilot flying an aircraft in HAVE. A set of additional environmental modalities were used to provide environmental representations for the computational models in the simulation which included both sensors and computational intelligent agents. The Viz-Sim subsystem in HAVE ensured that all the virtual environments for humans, agents and other computational models remained synchronised and consistent.

use by human pilots using the simulation. This manifested itself as a graphical rendering of the world displayed for the human pilot. The human centred modality was a lighted, visual representation of the world using a scene graph implemented using the Open Scene Graph [19] rendering engine. This was the view presented to the human operator. It also included a set of graphical effects (such as the addition of ground and cloud textures) which provided additional visual realism for the human, but had no explicit representation in any of the other environmental modalities.

The fourth design challenge was ensuring that the human and agent centred environments remain consistent and synchronised. This was accomplished by designing a bridge between these modalities, represented by the SimVizBridge package in Figure 2.4.

By addressing these design challenges, HAVE provides a single virtual environment with multiple modalities that allow for both agents and humans to interact in the same simulation while being provided with representations of the world that were suitable and accessible to them.

2.5.2 Environmental Modalities in HAVE

All simulation entities that have a physical presence in HAVE manifested themselves in the physical modality, which also included a collision detection and response capability. Not all simulation entities had a physical manifestation.

Teams of agents such as sides, forces and squadrons were explicitly represented in HAVE. The concept of a team of agents is an abstract social concept which does not necessarily have a physical locality. Furthermore in a military context, the relationships between teams of agents in terms of command and control (C^2) concepts need to be explicitly represented. Therefore a team modality that incorporated concepts of teamed agents, social structure and command and control concepts was explicitly represented in HAVE.

The electromagnetic environment was also represented in HAVE using a number of modalities. Entities capable of emitting radar signals (such as fighter aircraft with a radar) as well as entities which had a radar signature (capable of reflecting radar energy) were represented in the radar modality.

Components such as radars, radar warning receivers and radar jammers all interacted via the radar modality. Similar modalities were also available for other parts of the electromagnetic spectrum. These included visual, infra-red, laser and radio communication modalities of the virtual environment.

The visual and infra-red environments represented the world that agents could see using their eyesight or infra-red sensors such as a FLIR.[19] These entities in these environments were annotated with labels that were relevant to various agents in the simulation. These annotations were used by specific agents to determine the affordances for that entity and to ultimately make a decision about what to do next in the world.

Table 2.1 illustrates the set of annotation types that were attached to entities in the environment which agents could perceive. These annotations could be considered additional

[19]FLIR is a Forward Looking Infra-Red Sensor.

Tables 2.1: The primary types of annotations provided on sensor tracks in HAVE. These annotations are used by the pilot agent to reason about the entities it can perceive in the world.

Annotation Type	Example
Class	Aircraft
Role	Fighter
Status	Damaged
Object Type	F/A-18
Side	Blue

Tables 2.2: Annotation Types for Pilot Agents. As the pilot gets closer to a potential ground target, more information (or levels of detail) becomes available. The table indicates the types of labels/annotations that become available to the pilot as he/she goes through four stages of perception (Detection, Classification, Recognition and Identification).

	Class	Role	Status	Object Type	Side
Detected	●				
Classified	●	●	●		
Recognised	●	●	●	●	
Identified	●	●	●	●	●

information to that traditionally provided by sensor tracks such as the range and bearing to a specific entity.

2.5.3 Annotations on Ground Targets

The label and annotations that were available to the pilot agent for any given tracked entity depended on how close the fighter aircraft was to the entity. As the fighter aircraft approached a designated tracked entity, more information about the entity would become available to the pilot. For example, although a pilot might perceive a vehicle on the ground using the FLIR sensor, the pilot wouldn't necessarily know that it was a tank, whether it is an enemy tank or the specific type of tank until the entity was within a particular range.

The availability of different annotations for the fighter pilot was split into four separate stages which were range dependent. These were *Detected*, *Classified*, *Recognised*, and *Identified*. Table 2.2 indicates which annotations were available for each stage. These annotations are used by the strike fighter pilot agent to compute the affordances available with respect to each entity. Therefore, as a strike aircraft approaches a set of entities on the ground, the various sensors provide additional information in the form of annotations.

As the annotations with respect to a certain entity such as a tank becomes more detailed, the affordances the pilot perceives with respect to that tank also change. Different action possibilities become available the more of these annotations become available. In HAVE, these annotations are grouped into levels as shown in Table 2.2.

For example, if the distance to an entity is so far that it can only be determined to be a vehicle, then it is considered to be *Detected*. The detection of a vehicle does not afford it being destroyed. If on the other hand the vehicle is known to be a tank, is functional, the type of the tank is known and it is also known whether it is a friendly or enemy tank, then it is considered *Identified*. The affordances available to the pilot will obviously change as the level of information about the entities on the ground changes.

A number of ground based entities were modelled in HAVE. These included friendly, enemy, and neutral (e.g. civilian) vehicles as well as military and civilian infrastructure such as buildings. Models of infrastructure such as buildings included annotations following the same conventions for all entities as laid out in Tables 2.1 and 2.2.

Modelling military vehicles such as tanks and armoured personnel carriers served two purposes. First it provided potential enemy targets for the strike aircraft on CAS missions and second, it provided the vehicles for friendly troops, which included the JTACs which called in air strikes via radio.

The ground vehicles were modelled using a number of components. These included a vehicle dynamics model, a visual sensor model and a radio model for communications. The vehicle model could be controlled in one of two ways; either a human driver using an external steering wheel and pedals which was supported by the HAVE architecture, or via a driver agent.

The requirements for intelligent behaviour of the driver agent were relatively simple and hence was implemented using a relatively straightforward BDI based agent with only a few plans, such as the ability to follow other vehicles resulting allowing for the modelling of large convoys.

Additionally a simple JTAC model provided the ability for a convoy of vehicles to call in a supporting air strike should there be contact with threatening enemy vehicles such as tanks.

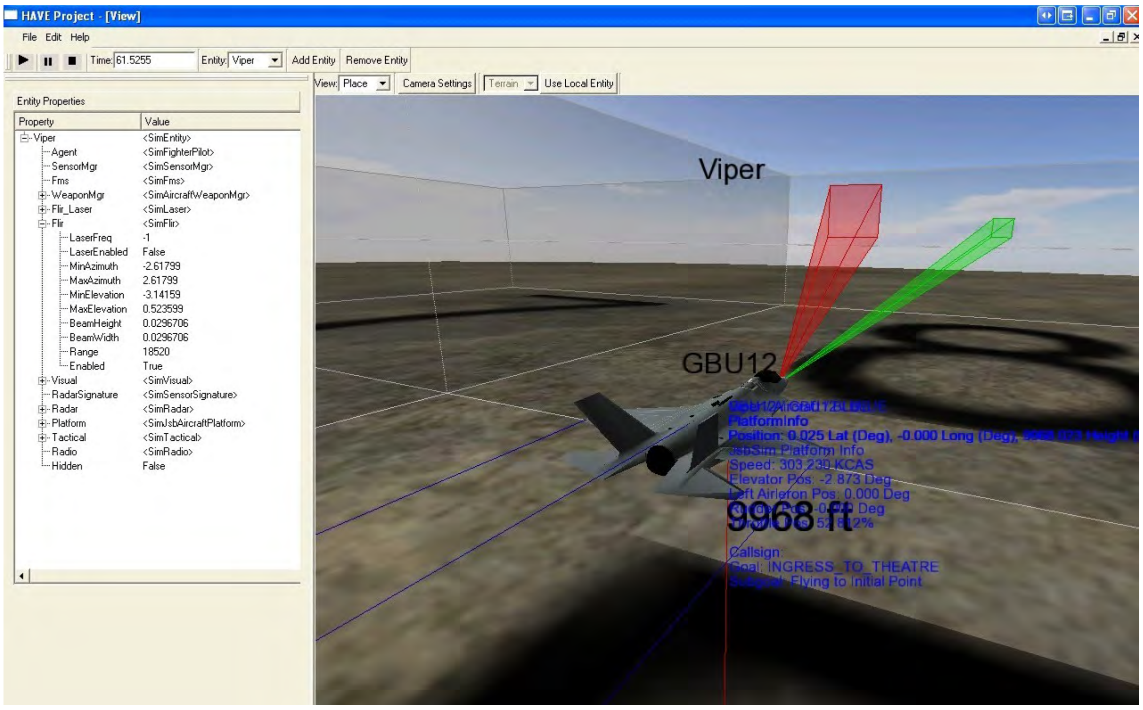
2.6 Modelling the Strike Fighter Aircraft

The strike fighter model in HAVE represented a modern multi-role strike fighter aircraft such as a F/A-18 Hornet with a weapons and sensor load-out configured for a Close Air Support (CAS) mission. Figure 2.7 is a UML class diagram indicating the various components which represented this aircraft. Each component had a unique role to play in a successful undertaking of a CAS mission.

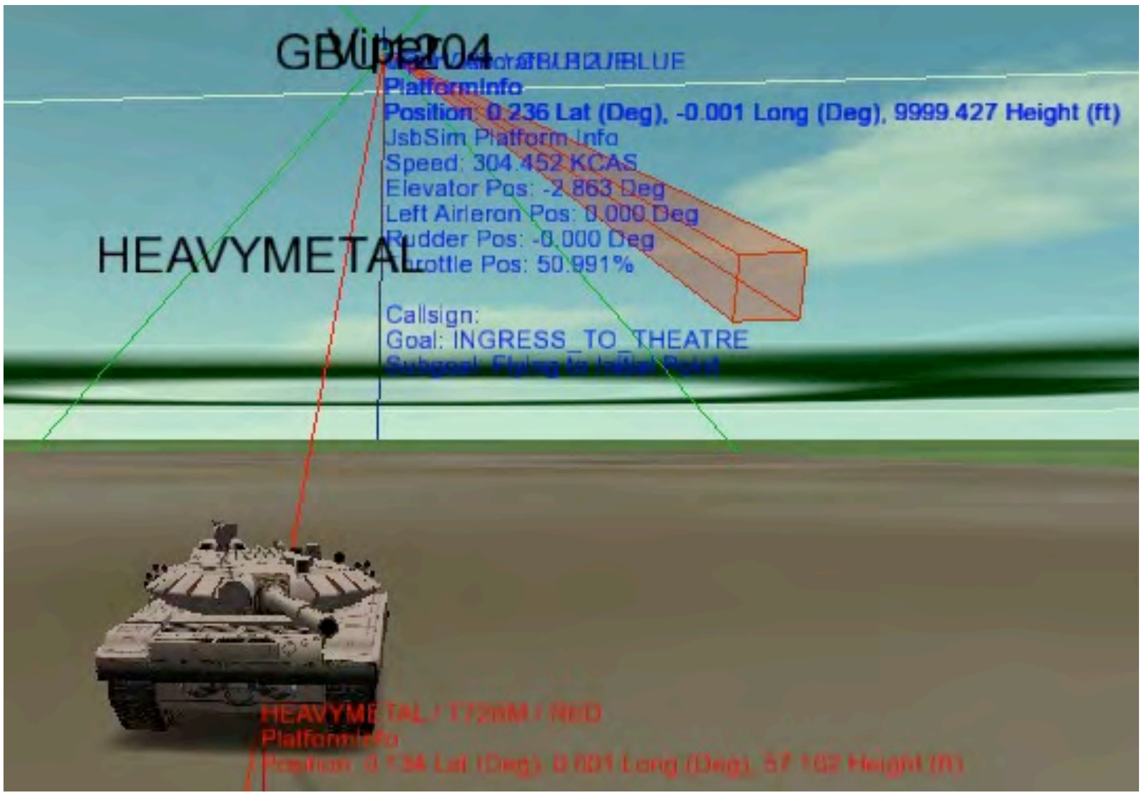
2.6.1 Aircraft Platform Dynamics

The aircraft platform model was composed of the flight dynamics model and flight control system.[20] The flight control system model was custom developed for HAVE and included

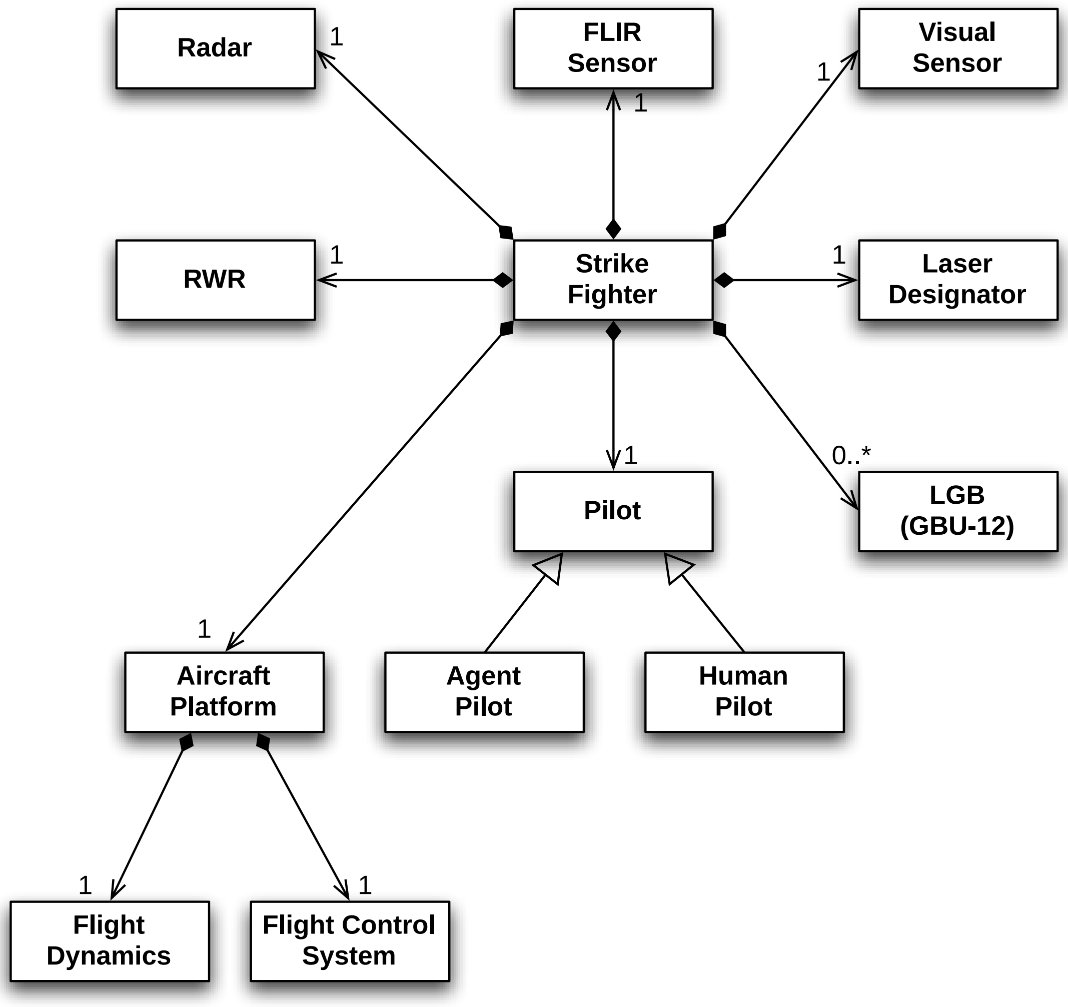
[20]An open source flight dynamics model known as JSBSim [13] was used in HAVE. The JSBSim based flight dynamics model allows for the representation of the flight characteristics of any aircraft using an XML based configuration file.



Property	Value
Viper	<SimEntity>
Agent	<SimFighterPilot>
SensorMgr	<SimSensorMgr>
Fms	<SimFms>
WeaponMgr	<SimAircraftWeaponMgr>
Flir_Laser	<SimLaser>
Flir	<SimFlir>
LaserFreq	-1
LaserEnabled	False
MinAzimuth	-2.61799
MaxAzimuth	2.61799
MinElevation	-3.14159
MaxElevation	0.523599
BeamHeight	0.0296706
BeamWidth	0.0296706
Range	18520
Enabled	True
Visual	<SimVisual>
RadarSignature	<SimSensorSignature>
Radar	<SimRadar>
Platform	<SimJsbAircraftPlatform>
Tactical	<SimTactical>
Radio	<SimRadio>
Hidden	False



Figures 2.6: (Above) HAVE visualisation of strike fighter aircraft on a CAS mission. (Below) HAVE visualisation of an enemy tank (callsign *HEAVYMETAL*).



Figures 2.7: UML class diagram which indicates the component models which were aggregated to represent a strike fighter suitable for a Close Air Support (CAS) mission in HAVE. The strike fighter model consists of a number of sub-component models. (1) Sensors: radar, radar warning receiver (RWR), forward looking infra-red (FLIR), and a human visual model. (2) Weapon systems: laser guided bombs (LGBs) and a laser designator. (3) Aircraft platform: flight dynamics model and flight control system (FCS). (4) Pilot model: which is an interface to either a human pilot or a computation intelligent agent pilot.

an auto-pilot mode and an interface that let it be controlled by a human (using a throttle and stick) or via high level commands sent to it by a pilot agent.

The flight control system (FCS) which was developed as part of HAVE served as an interface between various controllers and the underlying flight dynamics model. The controllers took on one of three forms.

Human Pilot: The human pilot controls the fighter aircraft in HAVE through a HOTAS (Hands On Throttle And Stick) or through a standard joystick. The flight control system then translated these into the appropriate low level control commands for the underlying JSBSim flight dynamics model.

Agent Pilot: The agent pilot sends high level commands to the flight control system which translated these into low level commands for the flight dynamics model. The agent pilot issues commands to the flight control system as a result of a decision making process which is discussed in more detail in Section 2.7.2.

Auto Pilot: The auto pilot can be considered as a very simple agent embedded in the flight control system. It allows both the human and agent pilots to issue a command (such as fly to a certain altitude) for the aircraft to fly automatically.

2.6.2 Sensor and Weapon Systems

A number of sensor models were used to represent the various sensors the pilot uses to perceive the information in the environment. These included a model of the radar for long range detection and tracking of air and ground targets, a radar warning receiver to detect radar emissions from threats, a visual model representing the agent's eyesight and a long-range infra red sensor.

The radar model represents the typical capabilities of a radar found on a fighter aircraft. This included the capability to scan and track targets using multiple air-to-air and air-to-ground modes. For the Close Air Support mission, the radar is used in one of the air-to-ground modes to find ground targets such as tanks.

The Radar Warning Receiver (RWR) is a sensor that detects air and ground based radar signals to determine if an enemy aircraft or enemy ground based air defence system obtained a lock on the strike fighter. The tracks from this sensor allow either the human or agent pilot to detect the direction of an enemy threat and undertake the required evasive action.

The radar is a strike fighter's primary long range sensor for detecting and tracking air and ground targets. The radar model in HAVE not only represents the maximum range and field of view of the radar, but also the scanning of the radar beam within these limits. It allows the human or agent pilot to set the radar in a particular mode which changes the scanning characteristics of the beam. The radar model also allows the pilot to modify specific parameters of the radar such as the radar range.

The radar model can operate in one of two primary modes. An air-to-air mode for tracking airborne targets and an air-to-ground mode for ground based targets. The radar can track multiple targets and can also be designated to lock on to a specific targets or to the nearest target. The radar model in HAVE produces information about the state of the radar and information about the targets it is tracking. This information is then used by the pilot to determine what to do next in the mission.

The radar model interacts with the radar environment in two different ways. Firstly it uses the radar environment to determine which entities in the radar environment which have a radar signature are within the radar's field of view. Secondly, the beam of each radar model in HAVE (when not in *Standby* mode) acts as an emitter in the radar environment. These emissions can be picked up by radar warning receiver (RWR) sensors on other entities (both air and ground based) to determine if they are being detected or tracked by a radar.

If the emissions happen to be coming from an enemy radar, and are being picked up by an aircraft's RWR it gives the pilot an indication of the threat level and allows for evasive action to be taken. Modern radar warning receivers typically indicate the direction in which a threat radar is being detected and can usually determine the type of radar the emission is coming from. This allows the RWR to produce a list of tracks indicating threats that are provided to the human or agent pilot.

The visual model represents the pilot's eyesight. It is used to identify air and ground targets. The FLIR (Forward Looking Infra Red) sensor is similar to the visual sensor except that it provides an infra-red image of the target to the pilot. It also has a longer range than the visual model and the ability to change the field of view to one of narrow, medium or wide. The FLIR is the primary means of identifying and designating ground targets for the pilot on a CAS mission. Once the pilot has identified a target with the FLIR, the laser designator is used to illuminate the target with a laser. The reflected laser light can be detected by a laser guided bomb, which enables it to guide on to the target.

While a variety of weapon systems are modelled in HAVE, the primary weapon used in HAVE is a laser guided bomb (LGB). This type of weapon is commonly used in close air support missions because it is suitable for use against smaller targets such as tanks and due to the accuracy provided by laser guidance. The laser guided bomb (LGB) model represents a free fall bomb which has a laser guidance kit added. The guidance kit can detect reflected laser radiation either from an airborne or ground based laser designator.

The laser designator on the aircraft is coupled to the forward looking infra-red (FLIR) sensor, so that the laser always points in the same direction as FLIR. This allows the pilot to use the FLIR to designate a moving target and have the laser designate which follows the direction of the FLIR. The pilot can issue three commands to the laser designator. The encoding command sets the laser code which the laser guided bomb should guide to, and the arming and disarming commands turn the laser on and off respectively.

Once the pilot has identified a target with the FLIR, he uses the laser designator to illuminate the target with a bright laser beam. The reflected laser beam can be detected by a laser guided bomb launched either by the pilot from the designating aircraft or from another aircraft such as the wingman.

A number of components are used to model a laser guided bomb in HAVE. These components are used to model the weapon's dynamics, the seeker, guidance and explosive systems. Interaction with the laser designator occurs through the laser environment. The laser seeker model on the weapon has access to the various laser reflections represented in the laser environment, but will only guide onto the laser reflection matching the one set by the pilot.

The detonation model used is relatively simple. When the weapon impacts the ground, all entities within a specified blast radius are destroyed. The blast radius is determined by the nature and size of the air to ground weapon.

Both human and agent pilots in HAVE can issue a number of different weapon commands. These include the ability to select a weapon type against a specific contact and to then launch it against the contact. When a weapon is selected against a specific target it continuously calculates whether a launch solution is available. A launch (or firing) solution typically occurs when the strike fighter aircraft has moved into a position close enough to the target where a launch of the weapon can successfully impact the target. In the case of a laser guided bomb, the weapon is launched against a target with a specified laser encoding.

2.6.3 Communications

Communication between fighter aircraft and other entities is critical for effective command, control and coordination functions during a mission. The modelling of radio communication in HAVE allows for the simulation of a number of aspects of a close air support mission.

- Strike fighter pilot communication with a controlling agency [21] providing command authority to proceed with a close air support or interdiction mission within a specified area of operations.
- Strike fighter pilot communication with a ground (JTAC) or airborne (FAC-A) controller in order to coordinate a strike on a specific target.
- Communication between a formation of strike fighters pilots in order to coordinate a search for a target or to coordinate a strike.

Communication between the various computational agents in HAVE (pilots, controlling agencies, JTACs) is accomplished through a model of a radio environment. Each agent that wishes to communicate with other agents in HAVE uses a radio model component. The radio model allows one agent to send a radio message to another agent on a specified channel.

The model allows each agent to specify which channels the radio should receive and transmit messages. The *SendMessage* interface specifies the radio channel on which the message is to be sent, which agent the message is *from*, which agent the message is going *to* and the content of the actual *message*. On reception of a message each agent is free to interpret the content of the message in any way that is suitable for the task at hand.

The radio environment provides the mechanism for radio messages to be sent and received by the various radio components being operated by the agents in HAVE. The radio environment performs a message routing function by only sending messages to specific agent radios that have been specified as the receiver of the message and which are listening on the specified channel. The radio environment also allows for the possibility of radio messages not being delivered due to physical effects such as propagation issues and line of sight issues such as terrain masking.

[21] A controlling agency may be airborne, ground based or on a naval vessel.

2.7 Modelling the Strike Fighter Pilot

The HAVE architecture allows for the strike fighter to be controlled by either a human pilot or a computational agent pilot. The interface to the strike fighter allowed the pilot model to receive all the necessary information from the various aircraft systems (such as platform state, sensor state and tracks) as well as having the ability to send commands to each of these systems.

For the human pilot the interface into the virtual environment consisted of a graphical rendering of a cockpit view with a Heads Up Display (HUD), sensor views such as a radar scope and FLIR display as well as the ability to control all of the systems using a throttle and stick. The pilot agent had the same capabilities as the human agent, except that there was a purely software interface into the virtual environment.

The focus of HAVE was to provide a common virtual environment for the strike fighter pilot, whether they be human or agent. This Section describes the pilot interface in HAVE from these two perspectives. HAVE provides a common virtual environment for both human and agent pilots using a number of different mechanisms.

First, the representations of the virtual environment in HAVE which the human perceives and which the agent perceives are based on a single environmental model that is adapted for viewing by humans or agents. The single virtual environment is represented as a scene graph and rendered for presentation to a human pilot. The graphical rendering of the environment (and associated instruments) allows the human pilot to perceive, understand, make sense of the world and ultimately make decisions and take actions in that world.

The same virtual environment is presented to the agent pilot through a set of environmental labels and annotations that are meaningful to the agent and allow it to make decisions and take actions in the world. Therefore, although the information about the world is presented to different observers (whether real or artificial) in different ways, the information is based on a single source.

Second, both the human and agent pilots are provided with identical interfaces to the model of the aircraft. Although these interfaces differ in form (for example a throttle and stick as compared to issuing a high level command), they are identical in function. This means that there is no action which the human pilot can take in HAVE that the agent pilot can't also undertake and vice-versa.

Third, at a more abstract level, affordances provide a common language in which to talk about perception and action in a virtual environment by both humans and agents. The behaviour of a pilot agent in HAVE can be explained and understood with reference to the affordances or possible actions the pilot agent perceived at any given time during a CAS mission. This provides a common language and reference point with respect to the affordances perceived by a real pilot undertaking a mission in HAVE.

2.7.1 The Human Pilot

HAVE was designed in a manner to allow for the control of a strike fighter aircraft to be easily switched at any time during a mission from agent control to human control and back

Figures 2.8: An air combat operations analyst flying a strike fighter aircraft in HAVE using a joystick interface.

again.[22]

The visualisation of the three dimensional world was implemented using a scene graph which is a common approach used not only in flight simulators but in many types of visualisation systems. This allowed the physical environment such as the sky, terrain and other entities in the world to be rendered in a manner which provided a level of immersion for the human pilot.

Additionally, visualisation of various aircraft systems were also provided to provide the human pilot with a similar level of information that is found on a real aircraft. These included a virtual heads up display (HUD) which provides information about the state of the aircraft, a radar display to show the pilot what contacts can be picked up by radar, a radar warning receiver display to indicate threats and a representation of a FLIR multi-function display that allowed the human pilot in HAVE to identify ground targets from large distances and which provided a cue with the laser designator.

The second aspect of the human pilot interface involved implementing mechanisms to allow the human pilot to fly the aircraft and control the various aircraft systems. While the HAVE application allowed the pilot to control all aspects of the aircraft using a combination of mouse actions and keyboard shortcuts, a joystick interface was also provided as illustrated in Figure 2.8.

The joystick interface allows the human pilot in HAVE to control the aircraft in a manner

[22]The same is true for control of vehicles such as tanks in HAVE. The control of any individual vehicle could be changed from a tank driver agent to a human driver through the use of steering wheel and pedals.

similar to a real strike fighter aircraft.[23]. This interface not only allowed the human pilot to fly the aircraft but allows the pilot to control the radar and FLIR sensors, control the laser designator as well as launch weapons.

2.7.2 Computational Agent Pilot

The agent architecture for the strike fighter pilot agent in HAVE was influenced by a number of different models of agent reasoning. These include the affordance based agent reasoning model presented in Chapter 5, the BDI model [120] of agent reasoning and Boyd's Observe-Orient-Decide-Act (OODA) loop model of military decision making.

In its simplest form, the pilot agent follows the *Perceive-Reason-Act* model, with percepts received from the environment, reasoning being undertaken and then atomic actions being enacted in the environment. However, the reasoning step combines elements from the BDI model, Boyd's OODA model and affordance theory in a novel way to reason about the world and to achieve the agent's goals. The pilot agent consists of the following components.

Percepts: All the raw information received by the pilot from the various aircraft systems and sensors. The pilot agent refers to perceptual information about an entity in the world as a *contact* (e.g. a radar contact).

Beliefs: The pilot agent has a series of beliefs. These beliefs come from a number of sources. They include pre-briefed information about the pilot's mission as well as beliefs about the pilot's own aircraft systems capabilities. Many beliefs are formed from the various percepts the agent is constantly being fed from the environment. These include beliefs about the state of the various aircraft systems and most importantly beliefs about the contacts in the environment obtained using the various sensors. Additionally, the pilot agent also has a set of second order beliefs. These are beliefs derived from other beliefs. For example if the pilot believes that a certain situation has arisen, then they might believe that they are under threat.

Intentions: The pilot agent has a library of possible intentions it can have. In many BDI based languages this is usually implemented as a library of plans. The agent's currently executing intention attempts to achieve a particular goal.

Affordances: Each pilot agent maintains a list of affordances with respect to every available contact. That is for every contact that the pilot can perceive, it maintains a list of possible actions it can take with respect to that contact.

Actions: The execution of the agent's current intention results in atomic actions being sent to the environment. The pilot agent has a library of actions it can take in the world. The actions result from the pilot agent's embodiment in the strike fighter aircraft and correspond to the various actions the pilot can take with respect to the various aircraft systems.

[23]Modern fighter aircraft are typically flown using a HOTAS (Hands On Throttle and Stick) A HOTAS consists a stick and throttle each with a number of switches which allow the pilot to control various functions.

Like all the other computational models in HAVE the pilot agent was executed by the simulation engine every time step. Each time step the pilot agent executes each of the four steps of an implementation of Boyd's OODA loop model as summarised in Figure 2.9. The four activities undertaken in each of the four steps (Observe, Orient, Decide and Act) are as follows.

Observe: The agent observes and perceives the world by processing the information obtained from the various sensors available. This includes radar, FLIR, visual, radio and RWR.

Orient: Using the observations about the various contacts in the world, the agent reasons about what affordances are available to it with respect to these contacts.

Decide: The list of available of affordances (with respect to specific contacts) in the current time step are made available to the agent. The agent must then decide if it should continue executing the current goal, or adopt one of the affordances (goal possibilities). It is not until the agent adopts a particular affordance that it becomes a goal that the agent has. Up until that point it is simply a possibility, meaning that the current situation affords the agent a course of action with respect to a particular contact in the world.

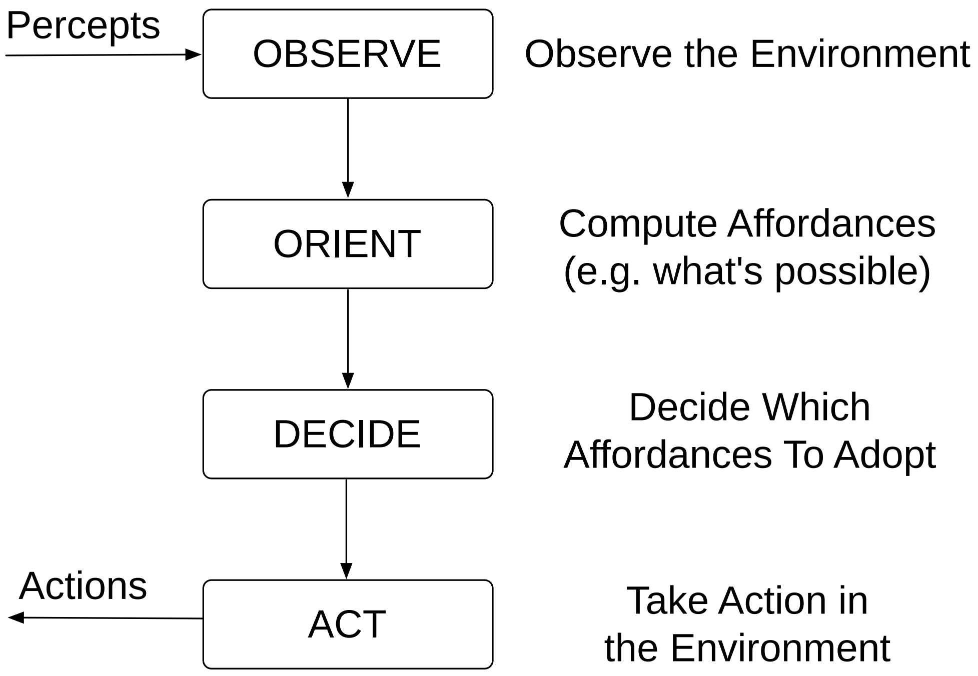
Act: The agent executes the next step in its current goal resulting a series of atomic actions in the world including controlling the aircraft, sensors, countermeasures, weapons and radio communications.

The details of how affordances were integrated into the design and implementation of the HAVE pilot agent is the most relevant with respect to the research presented in this thesis.

The first issue that needed to be resolved was to decide where the affordances were going to be computed in HAVE. The two locations considered were either in each agent or in the virtual environment. In HAVE the design decision was made to compute affordances inside the agent model itself rather than in the environmental model or another component of the simulation. This design decision was made for two reasons. First, too keep the introduction of affordance based agent reasoning into a complex simulation as simple as possible and second, to see if an implementation of an affordance based approach to agent reasoning was viable.

This meant that the introduction of the affordance concept could be kept localised to the agent model and not affect other parts of the architecture. There were a number of implications in making this design decision. The first was that this design is not consistent with the view in ecological psychology that affordances are not mental attitudes but rather arise from the interaction between agents and their environments.

The second issue that needed to be considered in the design of the pilot was to determine how to represent affordances. Each pilot agent in HAVE maintains a list of contacts. Typically this list of contacts represents entities that the pilot perceives using the various aircraft sensors. Associated with each contact, is a list of affordances or action possibilities that the pilot agent can undertake.



Figures 2.9: An affordance oriented interpretation of Boyd's OODA loop. This was the model of agent reasoning used for the strike fighter pilot in HAVE.

Internally this is implemented as a map [24] of contacts to a list of affordances with respect to each contact.[25] This allows for multiple affordances with respect to many different contacts. This approach also captures the relational aspect of affordances. The affordances are not simply arbitrary actions that the pilot can take, but they are action possibilities associated with specific entities in the environment which the pilot represents as contacts. This allows for different agents (in this case pilot agents) to have different affordances with respect the different entities in the environment.

For example if a pilot agent has four contacts, and has three affordances with respect to the first contact, none associated with the second, one associated with the third and two associated with the fourth, then the affordances will be represented as a map in the HAVE pilot as follows.

The complete set of affordances for agent i at simulation time is denoted by the symbol $\Phi_i(t)$. The contacts (other entities in the world) which agent i can detect with its sensors are denoted by the symbol c_j where j represented the j^{th} contact. An affordance agent i has with respect to contact (or entity) j at time t is denoted by $\phi^k_{ij}(t)$ where k denotes the k^{th} affordance, since it is possible for an agent to perceive more than one affordance with respect to each entity it can perceive. Square brackets [] are used to denote a list and the associative array or map is specified using a comma separated sequence of key/value pairs denoted by *key* : *value* enclosed in curly brackets {}. Therefore, for the above small scenario the complete set of affordances for agent i with respect to a set of four contacts at simulation time t is denoted by $\Phi_i(t)$ as follows.

$$\begin{aligned}\Phi_i(t) \quad = \quad \{ & \\ & c_1 \;:\; [\phi^1_{i1}(t), \phi^2_{i1}(t), \phi^3_{i1}(t)], \\ & c_2 \;:\; [], \\ & c_3 \;:\; [\phi^1_{i3}(t)], \\ & c_4 \;:\; [\phi^1_{i4}(t)], \\ \} & \end{aligned} \tag{1}$$

It is important to note that although the list of contacts the pilot agent maintains are primarily obtained via the various tracks provided by the sensors the pilot agent has access to, they do have a number of important features. The pilot agent may maintain information about a contact even if the track associated with a contact has been lost on a sensor. In other words, even if the pilot agent cannot actually see a vehicle or another aircraft does not mean that it will discard information about that contact, even if the information is out of date. This allows the pilot agent to have affordances with respect to entities that are not currently perceivable but which the pilot still may have beliefs about.

This also works with respect to contacts the pilot has that have not been obtained from the various sensor systems. For example, the pilot may know about a contact because it has been included as part of the mission brief or has received information about a contact over the radio from a wingman or ground based JTAC or some other controller. The design of

[24] A map is also known as an associative array, map, hash or dictionary.

[25] In HAVE the affordances for each pilot are implemented in C++ using an STL (Standard Template Library) `map`. While a `map` is implemented internally as a red-black tree in the C++ STL, similar data structures such as associative arrays or dictionaries can fill the same role.

the pilot agent in HAVE allows for the pilot to have affordances with respect to all these contacts which are not currently perceivable but may be in the future.

Another issue relating to the representation of affordances in HAVE was deciding what the actual affordances for the pilot agent should be. Initially it was considered that a library of all possible affordances would come from the list of all the atomic actions that the pilot could undertake in the world which were constrained by the embodiment provided by the aircraft systems. This would mean that the pilot's action possibilities would be fairly low level actions corresponding to commands to control the various aircraft systems such as setting the radar to a particular mode or flying the aircraft to a particular altitude.

While these types of action possibilities are necessary, it was found that they were not sufficient to capture the higher level and more meaningful possibilities that arise in the course of a mission for a pilot. That is, it was found that it was more useful and meaningful to think of affordances as not only possible atomic actions that the pilot could undertake in the world, but also to consider the possibility for a more complex series of actions. Traditionally complex series of actions are represented in agent languages as goals and can be implemented as plans and in BDI languages take on the form of executing intentions.

Therefore in HAVE, the affordances considered by the pilot are not only atomic exogenous action possibilities but also intention possibilities. This abstraction allowed the affordances in the agent to be designed at a level which corresponded to the type of affordances a human pilot might described.

For example, if a particular ground target afforded attacking, this is not necessarily a simple atomic exogenous action. Attacking a target may involve a complex series of actions involving positioning the aircraft, coordination with ground troops and other aircraft in the formation, the use of various sensors as well as the launch and guidance of a weapon. Hence attacking a target can be considered a possible goal or intention that the pilot might adopt. Thinking about affordances as intention possibilities is both a powerful and meaningful way of thinking about how an agent interacts and perceives the various entities in the environment.

The *Orient* and *Decide* steps in the agent reasoning model are the parts of the agent where the computation of affordances is focused. The role of the *Orient* step is to compute the affordances the pilot has with respect to each of the contacts and to populate the affordance map. The role of the *Decide* step is to evaluate all the affordances in the affordance map and to decide which affordance, if any to adopt for the next time step. This two step process clearly delineates the reasoning processes in determining what affordances are available to the agent and determining which ones to adopt.

In previous implementations of the OODA model in air combat simulations, the *Orient* step has been used to compute second order beliefs usually referred to as a situation assessment. The assessment of the situation made use of the pilot's beliefs and current intentions and generated second order beliefs such as whether the pilot was under threat. The computation of affordances in the HAVE pilot agent is consistent with this approach in that affordances are an assessment of the pilot's situation. The list of affordances available to a pilot agent tell the pilot what can be done in the current situation.

The affordances are also computed using the pilot's current beliefs and intentions. In fact the two most important considerations in the computation of affordances in the HAVE pilot

Example of Affordances in a Close Air Support Mission

1. The fighter pilot observes that the radar can detect a number of ground tracks. The track annotations inform him that they are all vehicles (Class = Vehicle) of some sort.
2. The pilot's *Orient* reasoning step informs him that each on of these vehicles affords identification. Associated with each vehicle track is the affordance *IDENTIFY*.
3. The pilot then decides to stop searching for vehicles and decides to adopt the affordance to *IDENTIFY* the entity closest in range.
4. The pilot has now adopted a new goal to *IDENTIFY* for the selected track of interest which results in setting the aircraft heading towards this track, reducing altitude and designating the FLIR sensor onto this target for a closer visual identification.
5. As the fighter flies closer to the track, the track state changes from *Detected* to *Classified* to *Recognised* and ultimately to *Identified*. The changes of track state indicate that more annotations from the virtual environment become available to the agent through its sensors. As the pilot gets closer to the track of interest he finds out that it is a tank that hasn't been destroyed, is of a particular type and eventually gets close enough to determine that it is an enemy tank. The *Orient* step in the agent reasoning model can now determine the track affords being attacked because it is an enemy tank.
6. The agent then abandons the *IDENTIFY* goal and adopts the affordance to *ATTACK TARGET*.
7. The execution of the *ATTACK TARGET* goal results in the pilot designating the laser onto the specified target and when within weapons range drops a 500-lb laser guided bomb onto the target, in an attempt to destroy it.

Figures 2.10: Example of Affordances during a CAS Mission agent are the annotations regarding the various properties of each contact associated with each contact and the pilot's current intention. The intentional nature of affordances makes this consideration critical. That is, the affordances computed for the HAVE pilot agent are not an arbitrary list of possible actions that the pilot can take at any time. Rather they are the possible actions the pilot can undertake that are conducive to achieving the pilot's current intention or goal.

When multiple affordances are available to the pilot agent, deciding which one to adopt can be a potentially complex computation. This is essentially a prioritisation process which may consider many different parameters. In HAVE the algorithm used to determine which affordance the pilot agent should adopt was straight forward. The two most important parameters that were considered were the range to a particular contact and the pilot agent's current goal. Therefore, closer contacts were given priority over those that were further away. If, after this process there was more than one possible affordance available, the first one was selected.

While Figure 2.3 shows an example of the affordances with respect to a single entity as perceived by different observers, HAVE focuses on modelling the affordances of a strike fighter pilot in a close air support mission. The other agents in HAVE (such as tank drivers) are modelled and implemented using traditional agent based approaches.

An example of how affordances are used by the strike fighter pilot agent in HAVE is shown in Figure 2.10. Although this is a scaled down example, it does illustrate a viable and interesting application of the concept of affordances for representing action and goal possibilities for a CAS mission in HAVE. Affordance determination is made easier through the use of annotations in the environment, reducing the need for a sophisticated human perceptual model.

This is important in military simulations where computational performance is a factor. Furthermore, an affordance based approach to agent design places a greater emphasis on environmental interaction and on the situated nature of agency. This is because the agent designer is required to think about the possible actions and goals that the various entities in the environment afford the agent under different circumstances.

2.8 Discussion

HAVE was developed to explore the interaction between agents and the virtual environment in the context a complex multi-agent simulation in the air combat domain. Specifically the aims were to:

1. Determine the viability of an affordance based agent reasoning model.
2. Investigate the use of multi-modal virtual environments.
3. Investigate the use of a common virtual environment in which both humans and agents can interact.

A number of important lessons arose from the development of the strike fighter pilot in HAVE. The first was that an affordance based model of agent reasoning could be incorporated with concepts and ideas from other models such as the BDI model of agent reasoning and Boyd's OODA model of military decision making.

A working strike fighter pilot agent that was capable of undertaking a simulated CAS mission in HAVE shows that an affordance based approach to designing these types of agents is viable. Furthermore, it allows operations analysts and military operators to conduct the activities of tactical specification, design, implementation, testing and the running of studies and experiments using the language of affordance and action possibilities. That is, an affordance based approach to designing agent reasoning in the these types of multi-agent simulations has benefits that extend beyond software engineering concerns.

Adopting an affordance based approach to agent reasoning also has the potential to impact significantly on the design of many parts of a multi-agent simulation. Like all software projects, the impact on design will involve assessing various trade-offs such as system requirements, performance constraints as well maintaining a balance between modelling fidelity and software engineering pragmatics.

In the case of HAVE, affordances were introduced as an internal agent mental construct. This meant that it was each agent's responsibility to compute which affordances were available to it, and to subsequently determine which affordances to adopt. While this is not entirely consistent with the view of affordances in the ecological psychology community where affordances arise from the interaction between agent and environment, there were solid justifications for this approach. This included the fact that it was possible to capture the essence of an affordance as an action possibility that an agent can undertake with respect to specific entities in the environment, while at the same time isolating the introduction of the affordance concept to the agents in the multi-agent simulation.

This is particularly important in existing or legacy military simulations where considerable resources have been expended to construct, validate and test many complex interacting computational models. In such cases introducing a new approach to agent reasoning which requires changes across many different subsystems may entail significant architectural design changes which bring with them the potential for greater resource expense and the introduction of additional project risk.

The design of HAVE also showed the advantages of a multi-modal environmental representation in a multi-agent simulation. Originally the design consisted of just two environmental modalities that were kept synchronised and consistent. One was designed specifically for graphical rendering and viewing by a human, and was implemented as a scene graph. The other was designed specifically to be accessible, understandable and meaningful to agent participants in the simulation.

Initially the focus was on designing an intelligent virtual environment that was appropriately annotated so that agents had access to similar information about the environment as would their human counterparts. However it was realised that the virtual environment accessible to the agent would have to be richer if affordances were going to be used as a mechanism for agent reasoning and interaction with the environment.

Whereas most simulations of real world phenomena focus on modelling the physical environment, the introduction of affordances requires a more sophisticated approach where more abstract modalities such as the social environment need to be included. This is based on the idea that the action possibilities available to humans or agents go beyond what is furnished by the physical environment and are heavily influenced by the other environments as well.

Designing a multi-agent simulation system such as HAVE in which both agents and humans could share a common virtual environment posed a number of challenges. The issue of computational model fidelity was one where a balance had to be struck between the operations research requirements and the requirements of providing an immersive experience to the human operator.

There are a number of areas in which the insights gained from the affordance based approach adopted in the development of HAVE, can be applied to other multi-agent simulations.

The first is the viability of an affordance based approach for thinking about how agents interact with their environments. HAVE demonstrated that an affordance based approach could be used in a complex domain such as the modelling of a close air support mission.

The second area is the important role in which environmental representation plays in these types of multi-agent simulations. A multi-modal approach to representing the virtual environment was adopted in HAVE because it allowed for multiple representations of the virtual world that were suitable for agents and for humans. Specifically it allowed for representations of the virtual world which were tailored to different types of observers, humans and computational models of humans (agents). The annotations on the entities in the world and the various modalities also made designing an agent that had an affordance based approach to reasoning much easier.

The third area relates to the design of the strike fighter pilot agent. The pilot agent in HAVE implemented an affordance based model of reasoning that was internal to the agent. That is, the affordances were not computed in the environment. One of the advantages of this approach is that it allows for affordances to be introduced into an existing multi-agent simulation within an agent model without significantly impacting the design of rest of the system. While this approach may not be a strict interpretation of affordances as espoused by the ecological psychology community it does allow for agent reasoning to be presented in the form of perceived action possibilities.

The fourth relates to legacy simulation systems. The design of HAVE has demonstrated that it is possible for existing simulation systems to be extended with additional environmental representations and the concept of affordance based reasoning to be introduced inside an agent without significantly affecting existing subsystems. For example, conceptually the graphical representation of the virtual environment in HAVE was no different from the virtual environment in many training simulators. By adding additional environmental modalities and ensuring they remained consistent with one another it was possible to integrate an agent oriented virtual environment without affecting the human oriented representation.

The development of HAVE represents a step in the direction of common virtual environments for both humans and agents. Ultimately the long term goal is for agents and humans to be able to interact in a common virtual environment either as adversaries or as team mates. The language of affordance and ecological psychology have a role to play as a common model of interaction.

Chapter 3

Literature Review: Situating Agents in Virtual Environments

"An important and extremely simple concepts that seems obvious in hindsight but is surprisingly new is the concept of ecological balance. This states that there exists a need for balance between the complexity of the brain, the complexity of the body, and the complexity of the environment.
...
People always forget about an agent's environment and its interactions with it, but is the most important consideration. "

— Dale Thomas [149]

3.1 Introduction

The purpose of this Chapter is to review the literature relevant to this research. However this Chapter does not set out to reference and review all the work relevant to this research in one place. Rather, references to the relevant literature appear in other chapters where appropriate. For example in introducing this research, work in the area of agent – environment interaction and military multi-agent simulation was described in Chapter 1. Similarly, a review of the nature of affordances from the ecological psychology literature relevant to the development of the model of affordance is described in Section 5.2 in Chapter 5.

This Chapter consists of two parts. The first part looks at the relationship between an affordance oriented approach to agent – environment interaction and three existing models of agency which have been adopted in this thesis. These are the situated view of agency from the field of situated cognition, the beliefs desires and intentions (BDI) model of agent reasoning and Boyd's OODA loop model of military decision making.

The second part of this chapter looks at related work. That is work where affordances or affordance like constructs have been used as a mechanism for developing intelligent agent behaviour in different types of multi-agent systems. In particular related work in the fields of multi-agent simulations, multi-agent systems, intelligent virtual environments, interactive entertainment and robotics are addressed.[26]

3.2 Affordances and Situated Agency

As one might expect, the most prominent field for research into affordance theory is that of ecological psychology; the study of how humans and animals interact with their environment. The work on affordance theory in ecological psychology follows on from the work of Gibson [54], and includes both theoretical and experimental research. Part of this research which is focused on the nature of affordances and is relevant to modelling affordances in multi-agent simulations is addressed in Chapter 5.

There is a significant body work associated with the use of affordances in industrial design which has been influenced by the work of Donald Norman [104]. Related to this work is the use of affordances in the design of user interfaces [3] to help improve human – machine interfaces. While some of Norman's views on the nature of affordances will be referred to in Chapter 5, affordance theory with respect to industrial design and human – machine interfaces is beyond the scope of this thesis.

However, the relationship between affordances and the wider cognitive sciences, and in particular the field of situated cognition is directly relevant. This is because the types of computational intelligent agents which this thesis focuses on are typically computational cognitive models of human decision makers situated in a virtual environment.

Developing credible computational models of human decision making and behaviour for use in applications such as military simulation requires at least some foundation in accepted theories of cognition. How strongly a computational models adheres to these theories will of course depend on the application domain and the nature of the problem being solved.[27]

Intelligent agents used in military simulations have been based on a number of different technologies and theories [114]. Two of the better known ones include SOAR [130], a cognitive architecture for general intelligence and the Beliefs, Desires and Intentions (BDI) model [120, 121] of agent reasoning. There have been multiple implementations of the BDI model including PRS [51, 50], dMARS [35] and JACK [74].

Both SOAR and BDI based approaches to developing agents have a proven track record of allowing for the design and implementation of sophisticated tactical behaviour for use in military simulations. However one of the problems with these heavyweight approaches to modelling intelligent behaviour is that they focus primarily on mental reasoning and provide a limited interaction with the environment for the agent. That is, they do not provide explicit support for reasoning about environmental entities or interaction – a problem which is difficult. This is problematic for a number of reasons.

[26]Although multi-agent systems, intelligent virtual environments and interactive entertainment applications (such as video games) are here treated separately there are times when these fields overlaps. For example many intelligent virtual environments and video games can be considered as types of multi-agent simulations.

[27]This is in contrast to some application domains such as in interactive entertainment (e.g. video games). In most video games it is not important if the theory and design behind the artificial intelligence in a character agent is based on a scientific theory. Rather, the emphasis is on providing an illusion of intelligent behaviour [20] for the purpose of entertaining the game player.

First, the idea that an agent is situated in an environment is at the centre of most definitions of agency [47, 15]. For example, Wooldridge defines an agent as follows.

> *"An agent is a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives."*
>
> — Michael Wooldridge [178, 179]

While BDI based agent languages such as dMARS and JACK provide many constructs for representing and reasoning about mental attitudes (such as beliefs, desires and intentions) and are suited at representing procedural reasoning, they provide limited constructs and infrastructure for interacting with a virtual environment. For example, although these languages typically provide a construct representing a belief, there is no language level construct which captures the agent's perception of the environment that allows for the representation of complex constructs such as affordances. This approach does have benefits as it allows for flexibility on the part of the multi-agent system designer in interfacing with whatever virtual environment is required.

However, this flexibility comes at a cost in that the resulting agents have a weak interaction with the environment. Percepts are fed in, some reasoning is undertaken and actions are sent back to the environment. This fails to capture the rich interactions that humans can have with their environment and subsequently computational models of humans should have with their virtual environment. It leads to agents that are less situated and resemble disembodied reasoners. Consequently the lack of a strong interaction with the environment means that agent reasoning which could be easier, becomes more complex and hence harder to design and program.

Second, modern theories of cognition espouse the view that intelligent behaviour is a result of the interaction between a human and the environment. Specifically situated cognition is a view that attempts to understand how the mind and intelligence work when it operates in an environmental context, rather than as a disembodied reasoner [26]. Unfortunately, despite the emphasis on agents being situated in many definitions, many computational implementations of agency provide for a limited environmental context. Although they provide the ability to model reasoning processes in a distributed manner, the interaction with the environment is not a primary design driver.

This is not an issue in many applications of agent technology. However, in applications such as military simulation where agents are often used as computational models of human decision making this is becoming increasingly important, especially as the types of behaviours that require modelling become more complex and the interaction with the environment becomes more sophisticated. In many cases it is the environment being modelled which becomes more complex and hence place additional requirements on the behaviour of agents in the simulation.

For agent systems that are used in such application domains, it is desirable that the underlying agent technology adheres to the principles of situated cognition rather than the traditional view of mind, which heavily influenced traditional artificial intelligence. Table 3.1 lists some of the characteristics of these two views of mind [133].

Tables 3.1: Concepts in the Traditional View of Mind vs Situated Cognition. Intelligent agents that are to be considered situated tend to be social, embodied, concrete, located, engaged and specific.

Traditional View of Mind	Situated Cognition
Individual	Social
Rational	Embodied
Abstract	Concrete
Detached	Located and Engaged
General	Specific

Affordance theory applied to multi-agent simulations has the potential to lead to agents that can be considered more situated. That is agents that are more closely aligned to the principles of situated cognition as listed in Table 3.1. This is because affordances are fundamentally a situated concept [26].

As the concepts outlined in Table 3.1 are quite general, there are many varied situated approaches to cognition. One approach that is relevant to agent systems is the work by Brooks in robotics [17]. In this approach the fact that an agent is situated in the world can be used to its advantage and be used to simplify the reasoning process.

"The world is its own best model."

— Rodney Brooks [17]

This means that it instead of an agent building its own internal mental model of the world and then reasoning about this model, it may be more efficient to let the world undertake the reasoning about what is possible, and then determine the outcome by observation. However in this thesis, the focus is on using ideas and concepts from the theory of affordances as a mechanism for designing more situated intelligent agents.

An affordance based approach to designing the agent – environment interaction is inherently a situated approach [26]. This is because the situated concepts of being social, embodied, concrete, located, engaged and specific directly influence the affordances or action possibility an agent can perceive in the world.

Affordances are inherently a relational concept. In a multi-agent system that means that they are also a social concept. An agent not only perceives affordances with respect to the objects in the world but also with respect to the other agents in the world. In many multi-agent simulations the possible actions an agent can take in the world are not only not only determined by the individual agent, but are often a function of an agent's social relationships with other agents. [28]

An agent's embodiment and the concrete physical constraints of the environment directly determine the type of affordances an agent can perceive in the world.

[28] This is particularly true in the case of military multi-agent simulations were decisions are made in the context of a complex social and organisational structure. The social interaction goes beyond adhering to the chain of command and includes the entire command, control and communications (C^3) formal organisational structure.

"The point is that you cannot expect a system to be intelligent if it is just a brain-in-a-box with no interaction with the environment. That is what is meant by the term embodiment. An agent needs to be inside the environment and interacting with it to be, or become, intelligent."

— Dale Thomas [149]

The information a human or animal perceives in the world and the actions they can undertake are constrained by their embodiment [4]. While this may seem obvious, many multi-agent systems consist of a set of disembodied agents which communicate with each other directly with no explicit model of an environment or agent embodiment. In application domains such as multi-agent simulation of military operations, explicit and concrete embodiment of agents is crucial. [29]

An agent's specific location and circumstances in the world will influence the affordances it can perceive. In other words, affordances are highly context dependant. In the situated view of cognition it is undesirable for agent reasoning to be separate from the activities of action and perception. Rather the context dependence that arises from being located in an environment is important. Furthermore the situated view promotes the idea that an agent's ongoing interaction and engagement with the surrounding environment is critical.

Finally, the decisions that situated agents make are heavily dependant on the specific circumstances which they find themselves in. An agent that is engaged in the world means that the affordances available to it will be constantly changing as the agent's position and circumstances change.

3.3 Affordances and BDI Agents

BDI based agent programming languages such as dMARS and JACK have been used in a number of multi-agent simulations of air combat, in particular the tactical behaviour of fighter pilots [70]. While specific agent programming languages have their own advantages and disadvantages when it comes to developing these types of simulations, there are advantages associated with having tactical behaviour defined using constructs from the BDI model.

One of the most significant advantages of using intelligent agents as models of human decision making in air combat simulations is that it allows operations analysts to interact with fighter pilots and other military subject matter experts in a way in which is not possible using traditional programming approaches. It allows for greater insight, ease of explanation and understanding into the tactical behaviour being performed by the virtual entities in the simulation.

There are a number of reasons why this is the case. The first is that agent programming languages based on the BDI model are amenable to being used as a common vocabulary

[29]The capability a military operator possesses and his or her ability to project military effect is not only defined by their cognitive and physical capacity but is enhanced by the equipment they use. For example, in an air combat simulation a fighter pilot is embodied in a fighter aircraft. The ability of the pilot to fight and effectively operate in the battle-space is determined and constrained by the sensors, weapons, airframe and other systems which make up a fighter aircraft.

between the operations analysts, the software engineers developing the simulation and the military subject matter experts such as fighter pilots whose behaviour and decision making is being modelled.

One of the reasons that this is the case is due to fact that the BDI model uses folk psychological constructs such as beliefs, goals and intentions. This means that tactical decision making during air combat can be discussed, interpreted, analysed and understood by all the various stake-holders in the multi-agent simulation. This is an extremely powerful tool not only in the requirements elicitation for the multi-agent simulation, but also during verification and validation.

The ability to explain the simulated tactical behaviour means that verification and validation process becomes more accessible to subject matter experts. Part of the reason is the accessibility to the concepts present in folk psychological models such as the BDI model.

In a similar manner, affordance based models of agent-environment interaction have the potential to play a similar role in making it easier to verify and validate tactical behaviour in multi-agent simulations. Affordances can be used to describe the specific behaviour undertaken by humans in their environment in terms of the action possibilities available to them. Multi-agent simulations which make use of affordances as a mechanism for modelling the agent interaction with the environment allow for agent behaviour to be explained in a similar fashion.

The ability to explain behaviour using terminology and language which can be understood by a domain expert becomes an invaluable tool in the process of verifying and validating these simulations.

In this thesis the model of agent – environment interaction inspired by affordance theory presented in Chapter 5 is presented in relation to some of the higher level concepts in the BDI model. While it does not formally extend Rao and Georgeff's BDI model or any particular programming language implementation, it nevertheless uses concepts from the BDI model such as intentionality. Agent reasoning models are often formally specified using a mathematical or logical formalism to allow for conceptual separation between the theoretical model and the implementation. For example, the Z formal specification language [140, 175] has been used to specify the BDI agent programming language dMARS [33], and the SMART agent framework [34]. Similarly the theoretical model presented in Chapter 5 will be specified using the Z notation.

3.4 Affordances and Boyd's OODA Loop

The second model which has been used in air combat simulation in conjunction with the BDI model is Boyd's OODA loop model of military decision making. This model has served a similar role to the BDI model in that it has been used to help explain tactical agent behaviour in complex multi-agent simulations to military subject matter experts [70, 71]. Colonel John Boyd [29] was a United States Air Force (USAF) fighter pilot who was not only influential in the design of modern fighter aircraft but is also known for the development of the Observe - Orient - Decide - Act (OODA) model of military decision making.

Boyd proposed that expert military decision makers continually (hence the use of the term OODA loop) undertake the following four steps during a combat.

(O) Observation involves observing the environment and processing all the information from the various sensors available to a military decision maker. For a fighter pilot this goes beyond perceiving visually what is outside the cockpit window. It also includes obtaining information from the various aircraft systems including long range sensors such as radar as well as communication from other pilots and commanders via radio.

(O) Orientation involves processing all the low level perceptual information and determining one's orientation in the world. This means bringing in all the information from all the sensors available to decision maker and forming a picture or an assessment of the current situation. In the case of a fighter pilot in combat this involves using information from all sensors to assess higher level questions such as are they winning, are they under threat and other higher level assessments. [30]

(D) Decision Once the military decision maker has oriented him or her self in the world, it is time to make a decision as to what to do. For a fighter pilot this can involve reasoning about what tactic to select based on the current situation.

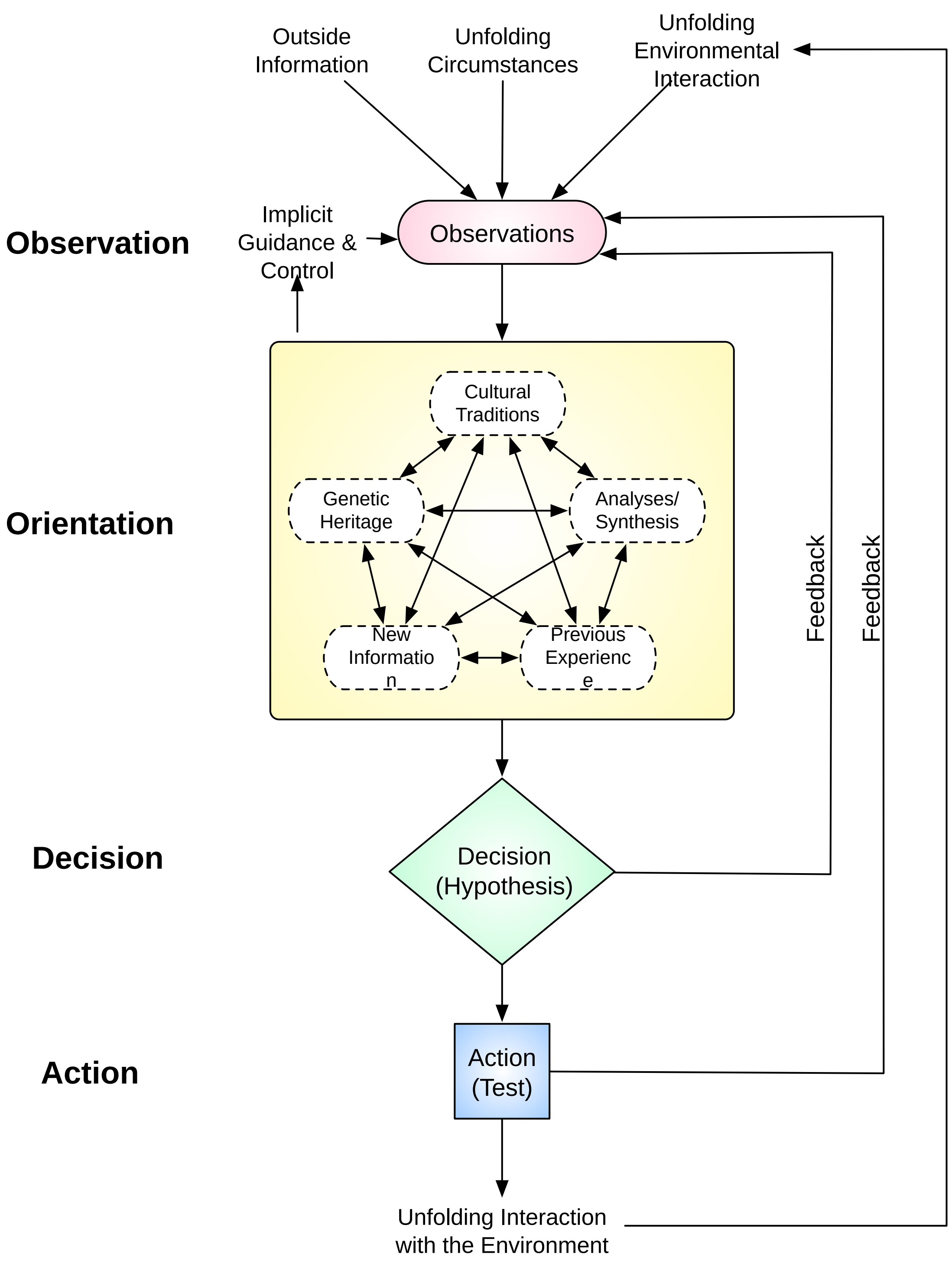
(A) Action Once a decision has been made, then it has to be acted upon and the corresponding actions need to be undertaken in the environment. For a fighter pilot this involves translating the selected tactic or course of action into exogenous actions that could correspond to aircraft manoeuvring, sensor and weapon employment.

A diagram depicting how Boyd viewed the OODA model is shown in Figure 3.1. [31] The diagram not only shows the four steps involved in decision making but also shows the complex dependencies, feedback loops and the various factors considered when a decision making is orienting him or himself. One of the most important ideas behind the OODA loop was the speed at which a decision maker executed the steps.

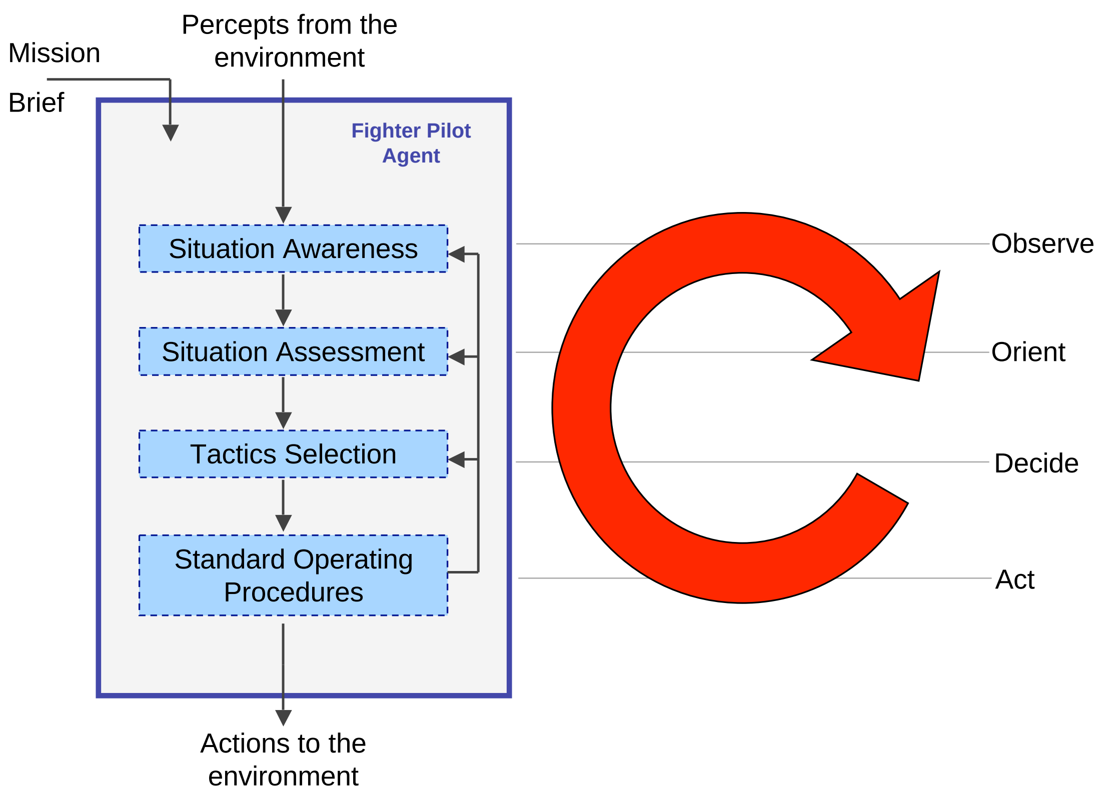
> *"Before Boyd came along, others had proposed primitive versions of an OODA loop. The key thing to understand about Boyd's version is not the mechanical cycle itself, but rather the need to execute the cycle in such a fashion as to get inside the mind and decision cycle of the adversary. This means that the adversary is dealing with out dated or irrelevant information and thus becomes confused and disoriented and can't function."*
>
> — Robert Coram [29]

[30]There are many similarities between the types of processing undertaken by an agent in the process of *Orientation* and the processing undertaken in multi-sensor fusion systems. Both involve bringing together information from various sensors and making higher level assessments regarding the current situation [69].

[31]This diagram of Boyd's OODA loop was adapted and drawn from the original sketched by Boyd, and modified by Chuck Spindey and Chet Richards. This diagram was adapted from the one appearing in Robert Coram's biography of Boyd [29]. It is important to note that Boyd did not officially publish an article describing the OODA loop. Rather, information about it appeared as a series of slides and briefings.



Figures 3.1: Adaptation of Col. John Boyd's OODA "Loop".



Figures 3.2: OODA loop in a Fighter Pilot Agent

The OODA loop as formulated by Boyd is complex. In many cases where it has been used in computational simulations it has been simplified. Figure 3.2 depicts how a simplified OODA loop was implemented for a fighter pilot intelligent agents in some air combat simulations.

In the model shown in Figure 3.2, the steps of *Observe*, *Orient*, *Decide* and *Act* have been interpreted as *Situational Awareness*, *Situation Assessment*, *Tactics Selection* and the implementation of *Standard Operating Procedures* (or Tactics Implementation).

When the OODA loop is considered as an agent reasoning model it can be considered to encapsulate two types of reasoning. The *Observe* and *Orient* steps involve reasoning about knowledge that the agent has about its environment (or the world) and hence can be considered as epistemic reasoning. On the other hand, the *Decide* and *Act* steps involve reasoning about procedures (such as tactical procedures) an agent undertakes in the world and can be thought about as procedural reasoning.[32] The OODA loop can then be considered a higher level agent reasoning model that captures both epistemic and procedural types of agent reasoning.

[32]Many BDI agent programming languages such as dMARS and JACK have been heavily influenced in their design by PRS or the Procedural Reasoning System, and hence are suited to procedural type reasoning.

In this thesis the OODA loop is used as a mechanism for framing an agent based model of affordance and for organising the different steps required of the decision making process. It is important to note that the full blown OODA loop as presented by Boyd is not being used in this thesis. Rather the four step model along the lines shown in Figure 3.2 is used for the framing of the affordance model presented in Chapter 5.

3.5 Related Work: Multi-Agent Simulation

The concept of affordances as a mechanism for interaction in the multi-agent simulation and in the wider multi-agent systems community is a fairly new one and hence there has been limited research in the area.

An affordance based approach implies that the environment plays an important role in any model of interaction. Research in multi-agent systems (MAS) has not focused as much attention on the environment in which agents are situated as it has on the modelling and designing the agents themselves as this quotes from Weyns suggests.

> *"The fact that functionalities of the environment are often treated implicitly, or in an ad-hoc manner, indicates that in general, the MAS research community fails to treat the environment as a first-class entity."*
>
> — Danny Weyns et al [171]

However recent efforts have attempted to bring the importance of the environment in multi-agent systems to the forefront with work in a number of different areas. This includes treating the environment as first class abstraction in multi-agent systems [58, 173], discussing the role in which the environment plays in multi-agent systems [85, 174], modelling and designing virtual environments specifically for multi-agent simulations [107, 86, 141, 12] and modelling the physical and social environment [8] amongst others.

The paper by Gouaïch and Michel [58] provides a step towards unifying the disparate views on how environments for multi-agent systems are dealt with. One the emerging ideas is that that representing the environment goes beyond just representing the physical or spatial world. The idea that the environment is inherently complex and can include abstract aspects such as the physical and social environments. The idea of a physical and organisational environment for multi-agent systems was proposed by Ferber and colleagues in the AGRE model [46]. It was further developed by Bandini [11] with a proposal for a model of the environment which is composed of several different layers (that can represent different conceptual abstractions) known as the *Multilayered Multi-Agent Situated System* (MMASS). This idea is very similar in concept to the idea of a multi-modal virtual environment used in the HAVE simulation in Chapter 2 and which is discussed in further detail in Chapter 4. A multi-modal or multi-layered environment is essential when representing affordance based interaction because affordances for each agent are dependant on the different aspects or modalities of the environment.

While there has been a recognition of the importance of the environment [171] in multi-agent systems, there has yet to be any accepted consensus or mainstream model of interaction emerge. As a result research into afffordance based approaches in multi-agent systems has often come from other fields, typically not directly related to the mainstream multi-agent systems communities.

In the area of geographical information systems or GIS, Raubal [125, 124, 123] and others [55] have used the concept of affordances to model a sense of place [80]. This has involved looking at the design of urban or public places such as airports through the use of multi-agent simulations. In these simulations a computational model of the airport as a virtual environment was developed and agents were used to represent travellers making their way through the airport. The traveller agents undertook various activities such as checking in and making their way to a departure gate.

The various facilities and amenities available at most airports (such as check-in counters, departure gates, arrivals halls, baggage claims) were not only represented in the virtual environment, but they were annotated with affordances relevant to the particular facility. For example, the baggage claim area afforded the picking up of luggage for arrivals passengers. Combined with annotated signs, the affordances allowed the traveller agents to find their way around the virtual airport and to conduct their business.

As shown by Raubal, one way of implementing affordances in a multi-agent simulation is to annotate objects in a virtual environment with the action possibilities that they can afford agents. However, the idea of annotating objects in a virtual environment is broader than affordance based labels or annotations that an agent can perceive. As described in Section 1.3 one of the problems with some virtual environments is that they are constructed primarily with the human participant in mind. As a result the underlying representation of the virtual environment is not always in a form that is easily accessible to computational agents. [33]

One of the significant limitations to Raubal's work is that the affordances can be considered as static annotations. That is, they are associated with specific objects in the virtual airport and represent the possible actions or *services* that these objects make available to travellers in the airport. However, they are not dynamic, context sensitive or change as the goals and intentions of the individual travellers change.

Other work has included that of Viezzer and Nieuwenhuis [165] who investigated developing an agent architecture that allows for agents to automatically acquire or learn affordance concepts. Similarly, Sequeira [134] looked at using the concept of affordance to allow for agents to learn about possible interactions with objects in the world based on previous interactions. The research involved developing a framework incorporating *smart objects* that help agents to identify possible actions they can take with these objects based on experiences they have had with other objects. This included categorising objects using basic contextual concepts such as shape, smell and fragrance.

The work of Cornwell and Silverman [30, 137, 136] has looked at incorporating affordances in multi-agent simulations as a way of modelling human performance.

[33]This problem is related to the problem of agents that search the world wide web for various types of information. The majority of the information on the web is available as HTML documents. While most information agents can read and process HTML, it is not an ideal situation. This is because HTML is concerned with the presentational aspects of information (such as formatting and colour), rather than the aspects that can be potentially meaningful to an agent. Hence, there has been a strong push to annotate information on the web in a form that is that is not only readable and accessible by information agents, but is also structured in a relevant and meaningful way.

Silverman has used affordance theory as but one model from a variety of fields in order to develop more realistic human performance simulations. Specifically, Silverman has incorporated affordances as one of many human performance moderator functions (PMFs). The approach taken is that...

> *"...each object in the simulated world executes perception rules to determine how it should be perceived by the agent and generates a list of actions (a_k) and affordances it can offer that agent."*
>
> — Barry Silverman [136]

Cornwell and Silverman specifically discuss an affordance based approach in the context of their *PMFserv* agent architecture. However, the model of affordance presented in Chapter 5 is described in an abstract manner, independent of any specific implemented agent architecture.

3.6 Related Work: Multi-Agent Systems

While this thesis is concerned with affordance based interaction in multi-agent simulations, there are number of areas in the broader field of multi-agent systems in which relevant research exists.

The field of Agent Oriented Software Engineering (AOSE) [76, 176, 14] deals with the software engineering of multi-agent systems and other software systems using agent oriented approaches. As such it has primarily dealt with agent oriented approaches to typical software engineering activities such as engineering methodologies [177], specification, design [105, 156], implementation, testing and verification and validation.

However, one of the aims of an agent oriented approach to software engineering is to help manage the complexity associate with the development of large multi-agent systems. As such, there is some commonality with the aims of the research presented in this thesis which deals with complexity in multi-agent simulation by considering the the interaction between agent and the environment. Therefore, issues of how the agent, the environment and their interaction is specified, designed, implemented and tested in multi-agent systems is relevant to this research.

Recent work in multi-agent systems and agent oriented software engineering has started to address the important role that the environment plays in these types of systems. For example the ROADMAP methodology [82, 81] explicitly accounts for the environment as a separate and distinct component of a multi-agent system. The work by Sturm [144] looks at a framework for evaluating agent-oriented methodologies, while the work Al-Hashel and colleagues [2] compares ROADMAP with other agent oriented software engineering methods.

In a paper addressing challenges and research directions in agent-oriented software engineering [180], Zambonelli and Omicini raise the environment and the agents interaction with environment, as an important consideration when modelling and engineering multi-agent systems (MAS).

"...it is clear that a MAS cannot cannot be simply reduced to a group of interacting agents. Instead, the complete modelling of a MAS requires explicitly focusing also on the environment in which the MAS and its constituent agents are situated and on the society that a group of interacting agents give rise to."

— Franco Zambonelli and Andrea Omicini [180]

In discussing the role that environments play in multi-agent systems, Weyns and Holvoet [172] state that if the interaction between agents and environments is to be taken seriously in the practical application of multi-agent systems, then these environments must be competently engineered. The approach taken by Weyns and Holvoet is one that not only promotes the environment as a first class abstraction, but also uses the notion of *artifacts* as environmental building blocks. Weyns and Holvoet define *artifacts* in this context as follows:

"...an artifact is a software entity designed to provide some kind of function or service that agents can use to achieve their goals."

— Danny Weyns and Tom Holvoet [172]

Weyns and Holvoet's concept of an artifact is not incompatible with the concept of an affordance in multi-agent systems as presented in this thesis. In fact, one might consider affordances as a special type of artifact, since they represent the goal directed action possibilities latent in the environment which help an agent to achieve its goals. Weyns and Holvoet classify artifacts into three types:

Resource Artifacts which mediate or provide access to a resource for an agent.

Coordination Artifacts which provide a coordinating function or service, supporting social activity.

Organisation Artifacts which serve an organisational or security function.

The idea of using coordination artifacts [34] as a mechanism for coordination and interaction in multi-agent systems was further developed by Ricci and Viroli [127] who describe them as follows:

"Coordination artifacts are runtime abstractions encapsulating and providing coordination services, to be exploited by agents within a given social context. They can be exploited as basic building blocks for designing and developing suitable working environments for heterogeneous multi-agent systems, supporting their coordination for collaboration or competition."

— Allessandro Ricci and Mirko Viroli [127]

[34]The use of artifacts in multi-agent simulations was motivated and inspired by social psychological theories such as Activity Theory (AT).

The work has treated coordination artifacts as first class entities that aim to facilitate a form of mediation amongst agents using them. The approach seeks to encapsulate strategies for constructing and ruling coordination activities. The work subsequently involves agents executing actions on these artifacts and a set of associated operating instructions, along with specific coordination behaviour specifications.

Several recent works have focused on developing a systematic artifact based approach for environment-based coordination in multi-agent systems. In addition to the work of Ricci and Viroli [127], there has been work in the area of generalising the notion of a coordination artifact to the more general artifact abstraction. Specifically there has been work in looking at artifacts for programming multi-agent systems [126] and designing and engineering multi-agent systems [166].

Unlike the aim of this thesis, coordination artifacts are entities that concern normative behaviour – and the effect these artifacts have on specifying and controlling coordinated action sequences – as opposed to the mechanisms that give rise to the possibility of specific action possibilities or affordances in the first instance.

The aim of this thesis is to characterise the mechanisms that make certain actions possible and available to agents as a result of interaction, from a relational (rather than a normative) perspective. with respect to the environment, other agents and other entities.

A further distinction between the coordination artifact approach to interaction and the affordance based approach used in this thesis can be found by looking at the difference between agents and coordination artifacts. In Ricci's and Viroli's paper [127] on coordination artifacts, agents and coordination artifacts are distinguished as follows:

- Agents are *goal-oriented* entities and hence agent models, languages and architectures are suitable for defining pro-active and autonomous behaviour of agents.
- Coordination artifacts are *function-oriented* entities. That is entities designed to provide some functionality or service.

While the concept of an affordance may facilitate coordination and may perhaps be modelled or be represented as a coordination artifact – which is different from a coordination artifact as described by Ricci and Viroli. An affordance does not provide a function or a service, rather an affordance provides an opportunity for action with respect to an entity that may provide a function or a service. The affordance concept provides the information to the agent as to what actions can be undertaken with respect to entity in the environment, rather than providing a function or service itself.

While this is quite a significant difference between the concept of a coordination artifact and the concept of an affordance, one important similarity is that both coordination artifacts and affordances as described in this thesis attempt to treat agent – environment interaction as a first class entity. While this thesis has focused on affordance based interaction, the broader problem of how to model of interaction as a first class entity has been dealt with by various researchers. For example Omicini, Ricci and Viroli [108] reviewed approaches for using coordination artifacts as first-class abstractions for engineering multi-agent systems. More broadly however, work done in the field of coordination models and languages [110, 167] has specifically looked at addressing the problem of treating interaction as a first class entity.

3.7 Related Work: Intelligent Virtual Environments

There has also been an effort to address the issue of agent accessible worlds in the community interested in the design and construction of virtual environments. Often referred to as *Intelligent Virtual Environments* or *Smart Environments* the work of Doyle [36, 37] and others [91, 90] has involved the annotation of objects and entities in virtual environments so that they are easily accessible and meaningful to the agents that are situated in these environments. Although this work does not always refer to the theory of affordances, it is related because affordances can be considered as a special type of annotation in the environment. One that is dynamic, tailored, context sensitive, agent specific.

This type of approach has a number of potential advantages. First, it raises the possibility of more flexible designs for multi-agent simulations, as it allows for agent behaviour to change by simply changing the labels and annotations in the virtual environment and not necessarily requiring any changes to the agent itself. Second, it means that agents have a much easier time trying to perceive and understand objects in the environment because it can be achieved through simple inspection of the labels or annotations. This means that the computational load on the agent can be significantly reduced by making the environment more intelligent.

The idea of sharing the computational intelligence between an agent and the environment is conceptually compatible with situated cognition, but it does raise some practical design issues. The most obvious of which is where to draw the line between agent and environment. One extreme is to design a multi-agent simulation where all the computational intelligence is in the agent, resulting in a complex agent and a simple agent. Another extreme is to design a very simple agent that lives in a complex intelligent environment. The alternative is to have a combination of the two where intelligent behaviour is the result of the interaction between the agent and the environment. The theory of affordances provides an appropriate and suitable mechanism for achieving such a balance.

Many of the examples described in the previous section involved offloading some of the reasoning demands involving perception and decision making from the agent and placing them in the environment. From a certain point of view, one might say that some of the *intelligence* that would normally live in the agent now goes in the environment. If the environment is being made more intelligent in some applications, it raises a legitimate research question as what are the appropriate components for locating intelligent behaviour in a complex system such as a simulation. That is, where do you place the *intelligence*?

In some application such as robotics, the real environment is a constraint and hence a robot agent must bear a large part if not all of the burden of perceiving and reasoning. On the other hand for many simulation and interactive entertainment applications it is possible to affect or augment the environmental representation to make it more amenable and accessible to agents, or in some cases more intelligent. The intelligent behaviour then observed in the system is a result of the interaction between an environment and an agent rather than having a purely heavy weight agent that is capable of doing everything.

"Put the smarts in the world, not in the AI"

— Jeff Orkin [109]

The above quote by Orkin [109] refers to a lesson learned by many agent (or game AI) programmers over many years, in that there are many advantages to developing an intelligent environment and making a simpler agent. One of the most important advantages of such an approach is that the intelligence is much more extensible by building smart environments as opposed to trying to build an agent which can handle every possible situation in advance.

Doyle [36, 37] argues that by placing domain knowledge in the environment and basic capabilities in the agent can result sophisticated and intelligent behaviours. In related work, Doyle and Hayes-Roth [39, 38] present the concept of agents in annotated virtual worlds. The idea is based on the concept of "knowledge in the world". Doyle and Hayes-Roth argue that if a virtual environment can be annotated with appropriate labels and annotations that explain the purpose, use of spaces and activities in the virtual world, then an agent that has access to these annotations will allow it to quickly and relatively easily exhibit intelligent behaviour subject to the constraints set by the annotated virtual environment.

Dautenhahn [31] discusses intelligent virtual environments (IVEs) which humans and agents can interact for the purposes of story-telling. The intelligent virtual environments of Dautenhahn are those that "contain some features where intelligence is *put into* and can be *attributed* to the environment".

Farenc et al. [45] also use the idea of an intelligent virtual environment for controlling virtual humans in urban environment simulations. Here in addition to a representation used for rendering, semantic information about various objects in the environment is also provided to the agents situated there. In this work, semantic information is provided to agents through *smart objects*, which are entities in the environment that for each possible interaction have pre-defined plans that can be used by the agents representing the virtual humans.

3.8 Related Work: Interactive Entertainment

Related to the work in intelligent environments is the work being undertaken for the development of agents and intelligent non player characters (NPCs) in the video game industry [99, 44, 75, 20]. The design and implementation of agents is often given the generic name *game AI* in the video games industry. Although driven by the need to immerse and entertain the game player, and not subject to the constraints of cognitive plausibility placed on it by the field of military operations research simulations, many game AI programmers are pushing the boundaries of agent-environment interaction and are increasingly adopting tools and techniques from academia as well as adopting novel methods themselves. While the use of affordances and related concepts for programming game agents is still in its infancy and not widespread, such concepts are being given serious consideration.

For example, in a collection of tried and tested tools, techniques and approaches for game AI programming published as *Game AI Programming Wisdom II*, an editorial on future promising techniques for game AI, explicitly mentions the use of affordances based techniques as a way of developing more intelligent characters.

> *"Smart terrain is the technique of putting intelligence into inanimate objects. The result is that an agent can ask the object what it does and how to use it.*

...
The use of smart terrain is enlightened by affordance theory, which claims that objects by their very design allow for (or afford) a very specific type of interaction.

— Steve Rabin [119]

One of the best examples of an implementation of an affordance-like concept in a video game is in *The Sims* [138]. The character agents *The Sims* interact with a virtual representation of the everyday world. They have homes, go to work and deal with every day objects such as household furniture, appliances and vehicles. In order for the characters to partake in flexible, intelligent and meaningful behaviour, the designers of the game annotated the every day objects in *The Sims* world with the actions that the characters can undertake with respect to the object. Coupled a representation of each character's state, this allowed for the game to produce the illusion of intelligent behaviour.

"Each object has a public interface that broadcasts its functionality to actors in the game. This is called advertising data in The Sims, and contains a list of possible actions and what motives they satisfy."

— Alex J. Champandard [22]

The broadcast of functionality to actors (agents or characters) in the game correspond to the types of actions the characters can undertake. While these have not been labelled affordances by the developers of the game, they are very similar to the types of affordances (as labelled annotations in the world) which are described in this thesis.

In many 3D action games such as first person shooters (FPS) both human and agent players need to find places in the environment which afford them a number of opportunities. This may include finding a place that affords hiding, or a location that is a good place to shoot from while still providing cover, or even a place in the world which affords ambushing an opponent. In order to provide this type of information to the agent characters in the game, a number of games have used an approach known as *terrain reasoning* or *smart terrain*.

This includes a family of techniques which involve annotating a game's virtual environment (or terrain) with action oriented information which can be inspected and reasoned about by the game's character agents. This is typically done either manually by a game level designer, or automatically in a pre-processing step. This means that these annotations are typically static for a particular virtual environment or game level and do not change dynamically as the situation changes.

One approach used by Liden [87] in games such as Half Life is to represent the locations on a map which an agent can get to using a set of nodes. A program then processes the nodes on the game map creating connections between nodes when it is possible for an agent to travel from one node to an adjacent node. The result is a traversal graph which is a representation of the environment which is suitable and accessible to agent characters and is at some level independent of the graphical representation of the world which is rendered for the human participant. In addition to node and waypoint traversing, the pre-processing steps in some games also considers the tactical advantages afforded by particular nodes that are situated in some part of the terrain or in a particular room in the game level. For example, a node in the graph which provides cover, or provides a good place ambush the enemy will be annotated as such.

In the game Quake III [161], this includes the annotation of action possibilities such as the ability to jump from one ledge to another. During the game play the agent's can simply inspect these annotations at make decisions about what to do next based on their current goals. One of the issues with this type of approach is the significant computation time required to generate these annotations.

In applications such as video games and military simulations where performance is a critical driving factor this means that the determination of these types of annotations cannot be done in real time and therefore have to be manually by a human level designer or computed in a pre-processing step. While these approaches are extremely powerful and allow agents to exhibit intelligent behaviour in the context of a game, they tend to be more suited to static environments which do not change over the course of game.

While these approaches to representing affordance-like concepts in virtual environments have been demonstrated to work, one significant limitation is that the affordances are static in a number of different ways. First, they tend to be fixed at design time and hence are not dynamic over the simulation run time. This is largely due to the pseudo real time computational constraints which are inherent in interactive virtual environments.

Second and more importantly they are usually implemented as static properties of the environment rather than relational properties which are dynamic and dependant on the particular agent. In the next section some of the practical issues associated with modelling and simulating affordances in multi-agent system are looked at in more detail.

3.9 Related Work: Robotics

Since this work is focused on multi-agent simulations in virtual environments, research in robotics is beyond the scope of this thesis. It is however important to acknowledge that there is active research in the field of ecological robotics [117, 42, 5].

This work attempts to apply ideas from Gibson's ecological approach to visual perception to designing and building robots that interact with the real world (as opposed to the virtual world which is the focus of this thesis). It includes work in applying the concept of optical flow to aid in the navigation [40] and locomotion of mobile robots [41]. Recent work in the field of ecological robotics has looked at applying affordance theory to aid in controlling robots [129, 132].

3.10 Summary

This chapter reviewed the relevant literature in two parts. The first part looked at the relationship between affordances and situated agency, the beliefs, desired and intentions (BDI) model of agent reasoning and Boyd's OODA loop model of military decision making – models of agency adopted in this work.

The second part looked at the use of affordances and affordance like constructs in a number of related fields, primarily multi-agent simulations, multi-agent systems, intelligent virtual environments and interactive entertainment.

In the following Chapter, the background issues associated with affordances and the interaction between agents and the environments which they are situated in are presented.

Chapter 4

Background: Affordance Oriented Interaction

"Every agent faces the task of making the best use of its environment. The environment contains a variety of goods and toxins, mixed in with a confusing host of more indirect clues...
Putting deliberate marks on the environment to distinguish what are for you its most important features is an excellent way of reducing the cognitive load."

— Daniel Dennett [32]

4.1 Introduction

In this chapter the issues associated with an affordance based approach to agent-environment interaction are discussed. The chapter begins by describing what is meant by the concept of affordance oriented interaction in multi-agent simulation. In particular, the justification as to why particular properties of affordance are considered in this thesis is presented.

The idea of affordances as annotations in a virtual environment which an agent can perceive is then described. The remainder of the chapter focuses on the implications on both the design of the virtual environment and the agent if an affordance based approach to this interaction is adopted.

4.2 Affordance Oriented Interaction

As mentioned in the introduction to this thesis in Chapter 1, the purpose of this research is not to develop a computational model of affordance that can be used to further study the affordance concept from an ecological psychology perspective. Rather the aim is to use the concept of affordance to provide motivation for ideas as to how one might better model the interaction between agents and virtual environments in multi-agent simulations. Hence an agent-environment interaction model that incorporates constructs, concepts and mechanisms which are affordance like can be considered affordance oriented.

While there is some broad agreement on what is meant by affordances, there is considerable debate as to their exact nature and their specific properties amongst the various communities which have adopted the concept.[35] A more in depth discussion on the debate on the nature of affordances is presented in Section 5.2 in Chapter 5.

However in order to discuss what it means to have an affordance oriented interaction between agent and environment, it is important to describe and justify why particular properties of affordances have been adopted in this research. This process was guided by a number of criteria.

1. How relevant and useful a particular property of the affordance concept was to modelling the interaction in the types of multi-agent simulations that this thesis is concerned with; namely military multi-agent simulations where the agents represent military operators such as fighter pilots.
2. How salient a particular property was to the overall concept of an affordance.
3. Whether a particular property was captured in previous representations of the affordance concept in other multi-agent simulations (such as those described in Chapter 3).

By using these criteria it was possible to select the most relevant and important properties of the affordance concept. One can consider an interaction between an agent and the virtual environment in a multi-agent simulation that incorporates these properties to be an affordance-like or affordance oriented interaction. The properties that can classify an interaction as affordance oriented include:

(1) Action-Oriented: The idea that an affordance is action oriented is fundamental to the entire concept. Affordances have been defined in many ways, but invariably they are typically referred to as the action possibilities [36] or the opportunities for action which they provide an agent. Most attempts at representing affordances computationally capture their action-oriented nature. This property is also important for military multi-agent simulation because the actions a military operator can take in the world are of interest.

(2) Meaningful: The idea that affordances are meaningful to the agents that can perceive them is a powerful concept. This refers to the idea that different agents can perceive different action possibilities with respect to the same entity. The possible actions that are available to each agent can be considered to be tailored as they represent unique possibilities that can be undertaken in the world which may not be meaningful to other agents. This property of affordances has not been addressed in a significant manner by previous work. Furthermore, one can imagine the important role in which this property plays in a military context. For example, an entity on the battle field will afford different things to combatants on opposing sides.

[35]These include the ecological psychology, industrial design and human-computer interaction communities.

[36]Refer to Definition 1 in Chapter 1 which defines an affordance as the action possibilities which an environment provides an agent.

(3) Relational: The relational nature of affordances has not been fully explored in existing work. This concept which says that affordances to do not exist in the environment or in the agent, but come about due to the interaction between the two is tied to the concept of affordances as meaningful action possibilities. The relational nature of affordances is explored in this thesis (in particular in Chapter 5) because it is considered a salient feature of affordances and can be used to capture action possibilities that arise during a military mission which come about due to complex relationships.

(4) Intentional: The intentional nature of affordances has not been explored in any meaningful way in existing computational representations of the concept. It is addressed in this research because it is considered a salient property of affordances. This idea says that the affordances an agent perceives in the world is not only dependant on the agent, but also on the agent's intentions. That is, the action possibilities an agent can perceive change depending on what the agent is trying to achieve in the world. This is highly relevant to military multi-agent simulation especially the ones where agent intent is explicitly modelled (for example in BDI agents). This is because the action's available to a military operator such as a fighter pilot change dramatically during a mission as the pilot's intent changes.

An additional property of affordance is included; one that is not dealt with in any significant way in this Chapter but is addressed in subsequent Chapers. This is the direct perceivability of affordances which has implications for how affordances are incorporated into the design a multi-agent simulation.

(5) Directly Perceivable: According to the view from ecological psychology, affordances are directly perceivable in the environment. That is, agents do not infer the possible actions they can take in the world based on perceiving properties of entities in the world. Rather, agents can directly perceive these action possibilities.

In the context of this research, these are the key properties which make an agent – environment interaction affordance-like or affordance oriented. These properties have been chosen because they represent they most salient aspects of the affordance concept, they have been largely not addressed by existing work and they are the properties most relevant to military multi-agent simulation.

How these properties manifest themselves in a computational model and ultimately a software design of a multi-agent simulation is of course subject to interpretation. The adopting of these properties from a modelling perspective can have a significant impact on the design of a multi-agent simulation.

In Section **??** a particular interpretation of these properties is adopted which looks at affordances as annotation in a virtual environment. Subsequently Section 4.4 looks at the implications for virtual environment representation when these properties of affordance are adopted.

4.3 Affordances as Annotations

Annotating virtual environments is a technique that has been used to make such environments more accessible to the agents that inhabit them. In the context of software architectures which implement annotated virtual environments, affordances can be viewed as a special type of annotation or label in the virtual world. Annotating virtual worlds provide a mechanism for implementing an affordance based agent-environment interaction in multi-agent simulations, particularly in existing or legacy systems.

4.3.1 Static Annotations

By and large the simplest and most common type of annotation in a virtual environment are those that are static in nature. These type of annotations are typically referred to as labels, tags, meta-data, keywords or notes.[37] Static annotations are typically associated with some entity in the environment, do not change over time and are accessible or readable by agents in the environment interacting with said entity. These static annotations typically describe properties that are inherent to a particular entity such as its type, shape and colour. The annotations are part of an ontology which is understood by the agents in the world. This means that the annotations should be in a language that the agent understands, and that their particular values are meaningful to the agent.

For example, in a training flight simulator, the underlying environmental representation may represent an aircraft model as a three dimensional mesh that is suitable for graphical rendering. While useful to the graphical renderer, this form of representation is not very suitable to a pilot agent that is situated in the same virtual environment. By annotating the aircraft with a label that describes it as an aircraft makes it useful to agents in the simulation that can perceive that aircraft. The label is not only accessible because it can be directly read by the agent, but it can be also be in a form that is meaningful to the agent.

One can argue that is amounts to *cheating* because the agent is being provided with information about the world which it would not have otherwise had access to without having to undertake some form of perceptual processing or complex reasoning to determine what that entity is. However, this is a legitimate and valid technique for a number of reasons.

1. Many applications do not need to model the low level sub-cognitive perceptual processing associated with object and feature recognition.

2. Directly accessing meaningful annotations about entities in the world that an agent can perceive leads to a simpler agent design, particularly the design of perception modules [72].

3. The concept is supported by the theory of direct perception [98] from ecological

[37]Annotation based approaches have become prevalent in a number of different areas such as semantically structuring web pages and tagging user generated content to assist search algorithms both on the web and on personal computers.

> psychology which says that information can be directly perceived [38] in the world by human and animals. Annotating virtual environments can be considered as a model of agent direct perception in a virtual environment.

Static annotations typically describe the properties of an entity in the world. An agent perceiving these annotations still has to undertake some sort of reasoning to determine what these properties mean and ultimately decide what actions are possible.

However, if in addition to annotations which describe basic properties, an entity is also annotated with more sophisticated static annotations such as the services and resource they provide, or the actions one can take with respect to them, then these can be considered as a rudimentary model of affordance.

Making the transition from annotations that describe inherent properties of an entity to annotations that the describe the actions an entity affords an agent in the world has two implications. First it provides an affordance based model of agent-environment interaction and second it provides the opportunity to add affordances to an existing or legacy virtual environment that already supports annotations in a relatively straight forward manner.

The caveat here of course this approach provides a rudimentary affordance model because of the static nature of the annotations. The affordances that a particular entity advertises or publishes are not dynamic and are the same for all agent observers. Although it is possible to capture some of the properties of an affordance with a static annotation, important characteristics such as their relational nature are not captured.

Despite these limitations, static annotations of entities in virtual environments with action oriented annotations provides a useful mechanism for implementing affordance based interaction that is (i) relatively easy to implement and (ii) is not computationally expensive.

4.3.2 Complex Annotations

Standard annotations of entities in virtual environments can be extended to provide a richer interaction with agents. By making the annotations more complex and sophisticated, the environment can be considered to be more intelligent. In order to better capture the concept of an affordance, it is necessary to extend the concept of a static annotation in a number of ways. Examples of how static annotations can be extended to capture the concept of an affordance are summarised in Table 4.1.

In addition to static annotations, virtual environments should support ones that are dynamic. While some properties of entities may not change, many other properties change over time. Dynamic annotations would allow an agent to perceive the properties of an entity that change over time such as an entity's position or other aspects of its state. These cover (both the static and dynamic) properties of an object which are perceivable by an agent and may include properties such as shape, colour, size, position, orientation and other properties characterising an entity's external state.

[38] A direct theory of perception is one in which information about the world is directly *picked up* as opposed to indirect theories of perception [18] where information from the world is *processed.*

Tables 4.1: Types of Complex Annotations

Types of Complex Annotations

Static Annotations Static annotations typically describe inherent properties of an entity. That is, they describe properties which do not change over time or as the state of the entity changes. For example an annotation describing an entity as an aircraft can be considered a static annotation.

Dynamic Annotations Dynamic annotations are annotations on entities in the world that change over time or as the state of the entity changes. For example an annotation describing the changing properties of an aircraft such as its speed or heading can be considered as a dynamic annotation.

Action Oriented Annotations Action oriented annotations are annotations which describe the type of actions that can be taken with respect to an entity. For example, affordances can be implemented in a multi-agent system as action oriented annotations.

Observer Tailored Annotations Observer tailored annotations are those that are specifically tailored for particular observers. The annotation associated with some entity changes depending on which agent is perceiving the annotation. Affordances are an example of this type of annotation because it is possible that two different agents will perceive different affordances with respect to the same entity.

Relational Annotations Relational annotations describe relationships between two or more entities in the world. This type of annotation is special because it is not associated with a specific entity. Affordances can be considered as a type of relational annotation.

Intention Oriented Annotations An intention oriented annotation is a special type of observer tailored annotation. It takes into account the intentions of the observing agent to present a tailored annotation. The intentional nature of affordances means that an affordance annotation in the environment can be considered to be an intention oriented annotation.

Introspective Annotations Describe or publishes properties of an entity which are typically internal, such as the mental state of an agent. For example an agent in the environment may annotate itself with its current intention.

However, for entities that are also agents, one can consider a situation where an entity might self-publish annotations about its own internal state. This may include annotations describing the entity's mental state or role in a team or other organisational structure.[39]

With respect to modelling affordances, one of the most important extensions that can be made to static annotations relates to the ontological nature of the annotation itself. While annotations regarding entity properties are no doubt useful to agents, action-oriented annotations are required to model affordance. It is the annotation describing an action possibility that allows a simple label on an object to be considered as an affordance.

Furthermore the relational nature of affordances suggests that if affordances are to be represented as annotations, that it makes sense to have relational annotations. That is, one can imagine an annotated virtual environment in which not only are the entities in the environment annotated, but also the important and relevant relationships are also annotated.

Perhaps the most significant extension is to allow for observer tailored annotations. That is, different agents should be able to perceive different annotations on entities and relations in the environment. The annotations perceived should be tailored to the individual agent observer.

This idea is important for capturing the idea that different agents will perceive different affordances even when perceiving the same entity in the world. Extending the idea of annotations in this way raises two important design questions.

1. How to tailor the annotations for the specific agent observer?
2. How to model the relational aspect of the annotation?

The question of how the annotations should be tailored for specific agent observers depends on the way annotations are intended to be used in a multi-agent simulation. For example in order for observer tailored annotations to capture and represent the concept of an affordance not only must the identity of the observer be taken into account, but also the observer's intentions and possible actions they can take in the world.

The second question is fundamentally about design. Originally annotations were discussed in the context of static labels that were inherently associated with specific entities in the environment. However, observer tailored annotations are essentially relational. This is especially true if they are being used to model affordances. Designers of multi-agent simulations have the option to either annotate specific entities in the environment or to represent the annotation itself as a relation between an agent observer and an entity being perceived.

The other design and implementation aspect to be considered is computational performance. One of the advantages of static annotations is that they are usually pre-computed and hence do not usually computationally prohibitive. As annotations become more complex computational performance may become a factor. This is especially the case when they are used to represent affordances.

[39]It was proposed by Heinze [67] that an agent that self-publishes or advertises its current intentions as annotations, allows other agents to infer those intentions by simply directly perceiving these intentions that were available as annotations. This was used as a mechanism for modelling intention recognition amongst agents.

4.3.3 Annotation Classification

It is important to note that the classification of annotation types described in Table 4.1 is by no means exhaustive or definitive. Rather they represent one particular interpretation of the affordance oriented interaction properties described in Section 4.2. Ultimately, the set of classifications that one chooses will of course depend on the specific requirements and the domain of the multi-agent simulation being considered. A number of comments can be made regarding each of the classification of annotation types.

The first comment relates to the overall stance of affordances as annotations. Annotations which are perceived in the environment by agents are consistent with the view that affordances can be directly perceived in the environment.

The second comment relates to static and dynamic annotations. The inclusion of annotations that are dynamic is an obvious choice to make. In the types of multi-agent simulations that are of interest in this research the dynamic nature of the environment is a key characteristic of the domain being simulated. While it is certainly possible that there are affordances that do not change over time and are the same for all observers, they represent an unusual case. The more interesting case is the one where affordances are dynamic and change depending on a number of factors. Hence, the inclusion of dynamic annotations is necessary for capturing an important property of affordances.

The third comment relates to the inclusion of action oriented, observer tailored, relational and intention oriented annotation classifications. The inclusion of these classifications correspond to capturing the salient properties of affordances described in Section 4.2, namely (1) action oriented, (2) meaningful, (3) relational and (4) intentional. These classification capture these properties of the affordance oriented interaction.

The fourth and final comment is on the naming conventions selected for the different annotation types. The nomenclature selected was motivated by terminology used in military operations research. Hence it reflects the types of terms an analyst might use to describe the a complex and dynamic battlespace. It is not suggested that the names selected should be mandated across all multi-agent simulations. Rather the application domain of a multi-agent simulation in question will heavily influence the annotation ontology.

For example in some application domains one might prefer the annotation types to be referred to as *permanent* and *transient* as opposed to *static* and *dynamic*. In the case of this work, the terms *static* and *dynamic* were not only chosen because they reflected the terminology used in the domain of military operations but also because they refer to how annotations vary over time. [40]

Annotations that do not change as the simulation progresses are *static* and are easier to deal with from the designer's perspective. On the other hand dynamic annotations are more difficult to design for because they not only change with the progression of time in a simulation but are also influenced by important events occurring in the simulation.

[40]An analogy can be drawn with the study of stationary bodies (statics) and moving bodies (dynamics) in the field of mechanics.

4.4 Environmental Representation

Introducing the concept of affordances to a multi-agent simulation has implications for how the virtual environment is represented. Affordances should make a virtual environment more accessible and meaningful for the agents that inhabit this environment.

In order for a virtual environment to be considered accessible and amenable to an agent, the design of the environment must possess a number of properties. These not only include the ability for an agent to interact with the environment with its sensors and actuators, but it also means that the form of the interaction is natural and meaningful to an agent.

In the same way as a software engineer should consider a set of quality criteria such as modularity, coupling and cohesiveness when designing any software system, the designer of a multi-agent simulation should consider an additional set of criteria and properties when designing a virtual environment that is to be populated by intelligent agents. How well the designer adheres to these criteria will influence how amenable and accessible a virtual environment is to an agent.

The representation of virtual environments populated by intelligent agents should be such that they posses a number of properties. These environments should be explicit, meaningful, relational, context sensitive, tailored, accessible, navigable, decomposable and support agent embodiment. Table 4.2 lists the most important of these.

While these properties are important, it is not expected that all virtual environments used in multi-agent simulations should necessarily attempt to implement them. The set of properties which are relevant and useful to a multi-agent simulation designer will of course depend on the specific application domain and the requirements of the system being developed.

Adopting an affordance based model of agent-environment interaction steers the environmental design into a direction favouring these properties. This is why adopting an affordance based approach naturally lends itself to designing virtual environments which are more accessible and amenable to agents. A more detailed description of some of these properties follows.

4.4.1 Explicit Environmental Representation

Perhaps the most important property that a virtual environment must possess in a multi-agent simulation is that it must be explicitly represented. While this may sound obvious, many simulation systems (in particular legacy ones) do not explicitly represent the environment.

For example in some multi-agent simulations, interactions between agents are represented using message passing architectures resulting in only an implicit representation of the environment. While this might be entirely appropriate in multi-agent simulations that only model activities such as communication between agents, it makes it very difficult to capture the rich situated interaction between agent and environment that is required many application domains.

Furthermore, the idea of an agent interacting with an environment is central to ecological

Tables 4.2: Qualities/Properties of an Agent Friendly Virtual Environment

Property	**Description**
Explicit	An explicit representation of the virtual environment should exist in a multi-agent simulation architecture. Interactions between agents situated in that environment are facilitated by the environment.
Relational	A virtual environment should not only capture and represent the entities in said environment but also the relations between these entitles. Agents should be able to perceive both entities and relations in the environment.
Meaningful, Context Sensitive and Tailored	The information an agent perceives in a virtual environment should be meaningful and useful to the agent. Agent percepts should vary depending on environmental context and situation. Each agent perceives the world in a different way. Information in the world should be tailored to each agent.
Accessible	An agent should be able to easily access the environment through its sensors and actuators.
Multi-Modal	An environment is made up of many modalities. These not only include the physical aspects, but also the abstract aspects such as the social and communicational environments.
Embodied Agency	A virtual environment should allow for agent embodiment.

psychology and to the theory of affordances. Therefore an explicit representation of the environment is an important requirement for a multi-agent simulation that incorporates the concept of affordance.

While it is perhaps possible to incorporate affordance concepts into a pure multi-agent system (one which consists solely of interacting agents), the lack of an explicit representation of the environment in which the agents are situated makes it difficult to represent the type of interaction captured by the concept of an affordance.

4.4.2 Representing Relations

Being able to understand and make decisions in the world is not only dependent on the object and entities that an agent can perceive but it also depends on being able to understand the relations between these entities.

A virtual environment should go beyond representing and modelling the various entities which are situated in it. A richer virtual environment should also represent the relations between these entities. Furthermore, an agent should not only be able to perceive the entities in the world but also the relations between entities.

Representing relations in a virtual environment means that it makes it easier to capture and represent the concept of an affordance. This is because affordances are inherently a relational concept – between an agent and an entity in the environment.

4.4.3 Meaningful, Context Sensitive and Tailored Perception

The information an agent perceives in the world should be meaningful, context sensitive and tailored to that agent. A meaningful environment means that the perception of environmental information which the agent has access to is in a language the agent understands.[41]

The information available to the agent should also change with respect to the situational context. The current situation makes a difference to the way humans perceive the real environment. This should also be true of how agents view the virtual environment under differing situational contexts. Similarly, affordances can change depending on situational context and hence a virtual environment which can present different information to an agent depending on the situation is suited to representing affordances.

Additionally agents should be able to perceive information about the environment that is tailored to them, meaning that although two agents may be perceiving the same entity in the world, they may not necessarily be perceiving the same information about that entity.

Introducing affordances to a multi-agent simulation requires the corresponding virtual

[41] At some level this is a question of matching agent perceptual ontologies and is heavily influenced by the application domain. For an agent to be able to perceive entities, labels, annotations and events in a language and terminology that it can understand makes it easier for the agent designer. It also has practical implications because it impacts on the design and implementation of an agent's perception module and consequently impacts on computational performance.

environment to be at some level meaningful, context sensitive and tailored. By their very nature affordances are meaningful, context sensitive and agent tailored because each agent perceives the action possibilities which are relevant to it, in the current situation and in the form of its own available actions.

4.4.4 Accessibility

Making a virtual environment accessible to an agent can mean a number of different things. At one level it means that the environment should provide suitable interfaces to allow the sensors and actuators of various agents to interact with the it. More significantly, accessibility refers to the form of the information which comes from the environment to the agent and how suitable it is for being processed by an agent reasoning engine.

Even if the environmental structural representation was in a form that was suitable for processing by an intelligent agent, the information must also be ontologically compatible with the domain ontology that the agent understands. Therefore, there are three aspects to providing accessibility to virtual environments by intelligent agents.

Interfacing Primarily a software design issue, a virtual environment can be made accessible if it provides appropriate and easy to use interfaces for agents to facilitate both perception and action.

Structural Representation How the underlying structure of the environment is represented can make a large difference to how easy (or hard) it can be for an agent to perceptually process information from the environment.[42]

Ontological Compatibility When both the agents and their corresponding virtual environments make use of the same ontology then accessibility between the two components becomes easier. It also means that affordances can be represented in a common language and do not need to be translated from one form to another.

4.4.5 Multi-Modality

The representation of the virtual environment should be inherently be multi-modal. That is, it is not just the physical environment that should be represented in the simulation architecture but also other environmental modalities which are relevant to the application domain.[43]

[42]For example a scene graph is often used to represent and structure virtual environments that are graphically rendered (with flight simulators being a good example). A scene graph typically represents the scene in a hierarchical manner with nodes used to represent physical entities as well as graphical operations such as transformations and lighting. However this structural representation does not represent other aspects of the environment which may be relevant to an agent, such as the structure representing the social and organisational environment.

[43]Here an environmental modality is used to refer to various aspects of the environment which an agent might interact with. The term environment is used in its most broadest sense, meaning that it goes beyond what is physically present but also includes abstract aspects of the environment (such as the social environment) which are relevant to humans and agents making decisions.

This not only means environmental modalities such as physical, electromagnetic and auditory environments, but may also mean more abstract environmental modalities such as social, organisational and communication structures. It is important to include all the relevant modalities because the humans which the agents are representing in the simulation make decisions about which actions to take based on all of these environments.

The affordances that humans perceive in the world are not only determined by the physical layout of entities they perceive but are also dependent on the social and organisational environment in which one is situated. For example, the possible actions an agent can take may be constrained by their role in a social or organisational structure. This is particularly true in a military context where the command and control structure plays an important role in the possible actions a military operator can take.

Figure 4.1 shows an example of a multi-modal environment that represents different modalities suitable for some military multi-agent simulations similar to one used in the Human Agent Virtual Environment (HAVE) describe in Chapter 2. The virtual environment is represented by a number of different modalities. These include the physical, radar, infra-red, visual, communications, tactical and the team/social modalities of the environment. Not all entities and agents in the simulation will necessarily manifest themselves in all the modalities. For example, an entity such as a team may not necessarily have a physical manifestation and hence may not appear in the physical modality.

Each modality should also capture the relevant relations that exist. For example, the physical modality may capture the physical relationships between entities, the communications modality may capture the relationships between agents that can communicate with each other, while the team modality may capture the various team and social relationships. It is important to capture all these concepts because they are required in determining what affordances are available to each agent.

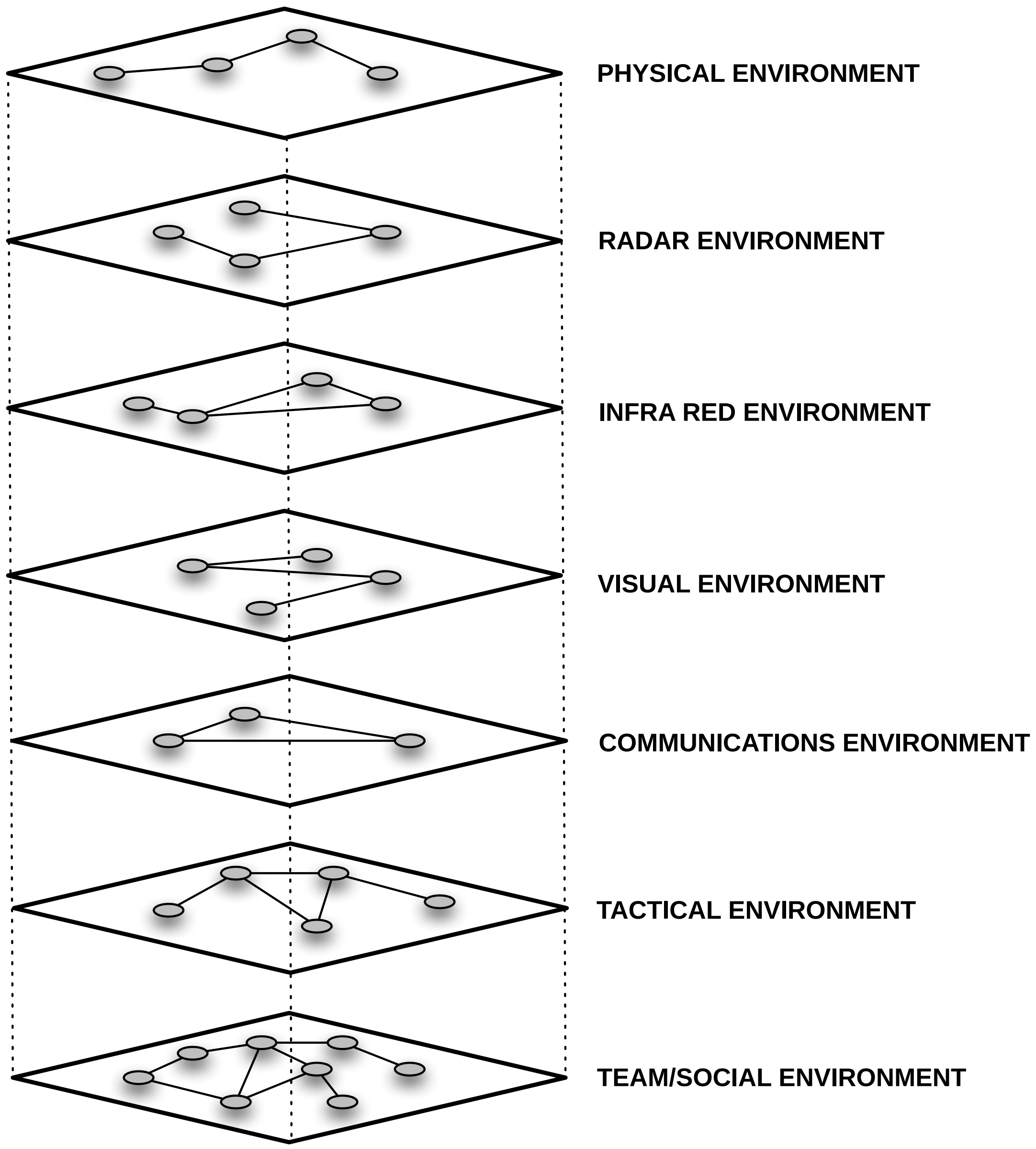
4.4.6 Agent Embodiment

A virtual environment should support the embodiment of agents. Traditionally many agent theories, architectures and languages have focused on modelling agent reasoning. When agents are used as cognitive models of human decision making as they are in military simulation, the agent reasoning model represents the *mind* of the human being modelled, and it is this component of this system which typically results in an agent being labelled as intelligent.

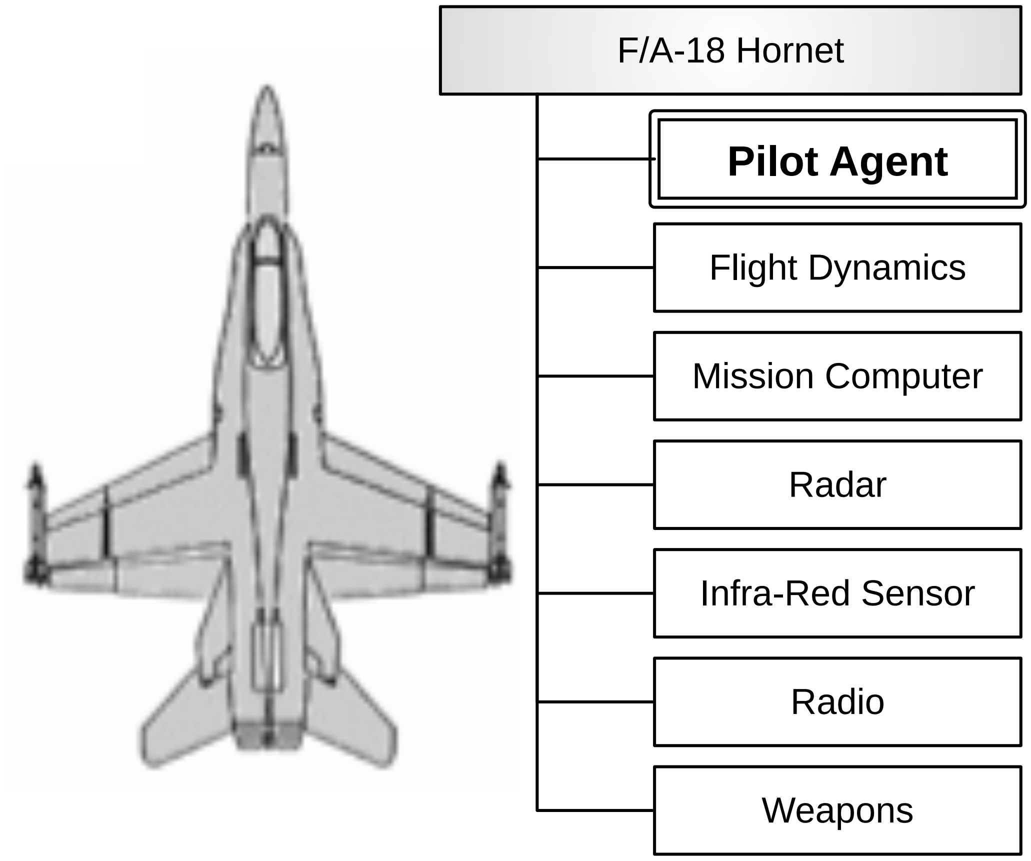
However for agents to be situated in an environment, to perceive and act in that environment, they must be embodied. Embodiment not only provides an agent a presence in the environment but also provides the agent with the ability to perceive the world through its sensors and act in the world through its actuators.

In some air operations research simulations (such as the HAVE multi-agent simulation in Chapter 2), the embodiment of a fighter pilot agent is represented by the pilot's aircraft, sensors and associated weapons systems as illustrated in Figure 4.2. It is important to note that embodiment in this case does not necessarily mean including a physiological model of the human body. [44]

[44]In many military multi-agent simulations physiological effects are not modelled. This is particular the



Figures 4.1: Representation of a Multi-Modal Virtual Environment. The nodes represent various entities in the world (such as objects, agents and teams). The arcs represent relevant relations in a particular environmental modality. For example an arc in a team modality might represent that two agents are on the same team. Not all entities manifest themselves in all environmental modalities. For example an entity representing a team of agents does not have a physical manifestation and hence will not appear in the physical modality.



Figures 4.2: Example of an embodied fighter pilot agent in a simulated F/A-18 Hornet fighter aircraft. The pilot agent does not interact directly with the virtual environment. Rather it is the agent's embodiment provided by the model of the fighter aircraft that allows it to perceive the environment and take actions in it. An agent's ability to perceive affordances in the world will be influenced and limited by the type of actions it can take in the world which are defined by the capabilities provided by its embodiment.

For example, in an air combat simulation a fighter pilot agent would be embodied through computational models of the aircraft platform, various sensors, communication systems, countermeasure systems and weapons. All these models allow the pilot to interact with the environment either by facilitating a sensing purpose or by allowing exogenous actions. Without representing all these components and their interface into the virtual environment, the agent reasoning model used to represent the pilot's tactical decision making is disembodied and non-situated.

The concept of embodiment is also central to the idea of an affordance as an action possibility. The possible actions an agent can take are directly dependent on the agent's embodiment. What actions a fighter pilot agent can take will depend on the aircraft, sensors, weapons and other systems. Affordances cannot exist independent of an agent's embodiment. What affordances an agent can perceive will depend not only on the agent's sensors but also on the agent's capabilities which are essentially defined by the agent's embodiment in the virtual environment.

4.5 Agent Representation

A decision to adopt an affordance based model of interaction also has the potential to impact on the design of the agents in a multi-agent simulation. There are two aspects of agent representation and design which are impacted by an affordance based approach. The first aspect is the design of the agent interface to the environment. The second aspect is the internal design of the agent which is usually referred to as the agent reasoning model or in the cases where agents are used to represent human decision making, the agent cognitive architecture. The issue of how to incorporate affordances into existing agent approaches raises a number of questions.

- How does one design an agent which can directly perceive affordances in the environment?
- How does this then fit in with the traditional *Perceive-Reason-Act* model of agency?
- How are affordances represented internally by an agent reasoning model?
- How can affordances be integrated into existing agent reasoning models such as the BDI model or Boyd's Observe-Orient-Decide-Act (OODA) loop model of military decision making?

The answers to these questions of course depend on the type of multi-agent simulation being developed and the corresponding application domain. These questions are subsequently answered from the perspective of modelling affordances in a multi-agent simulation in Chapter 5.

case in operations research simulations where the focus is typically on understanding the tactical outcomes of a military mission.

4.6 Summary

This chapter looked at some of the higher level issues associated with taking an affordance oriented approach to agent – environment interaction.

The chapter began by presenting the view that affordances could be viewed as a type of annotation that could be applied to entities in the virtual environment. In particular it made the distinction between simple, static annotations and more complex annotations, whose characteristics are more suitable to capturing the essence of an affordance.

The chapter then moved on to describing issues associated with environmental representation. In particular a number of qualities that a virtual environment needed to possess to make it agent friendly were then discussed. Specifically these qualities and properties were discussed in terms of affordances. The introduction of an affordance oriented approach suggested a set of properties for virtual environments which made them more accessible and amenable to interaction with computational agents.

Finally the chapter described some of the issues associated with agent representation in an affordance oriented multi-agent system. The issues and questions raised in Section 4.5 are addressed in Chapter 5 which looks at the specifics of modelling affordances in multi-agent simulations.

Chapter 5

Modelling Affordances

"The world of physical reality does not consist of meaningful things. The world of ecological reality, as I have been trying to describe it, does. If what we perceived were the entities of physics and mathematics, meanings would have to be imposed on them. But what if what we perceive are the entities of environmental science, their meanings can be discovered."
— James J. Gibson [54]

5.1 Introduction

This Chapter tackles the necessary elements toward developing a computational model of affordance, that facilitates integration with agent programming techniques utilised in simulations. In one sense the model strives to bridge the gap between ecological psychology and software engineering, on another it tackles the design and architectural elements required for modelling the interaction between agents and their environments.

One of the challenges in developing such a model is the lack of precision and the ongoing debate on the nature of affordances in the ecological psychology community [84, 78, 158, 159]. For example, how does one capture affordances as meaningful, perceivable and discoverable things in a computational model in the manner in which Gibson refers to them in the above quote?

This Chapter begins by drawing on the discussion on the nature of affordances in attempt to capture the properties of affordances which are relevant to multi-agent simulation. Based on this information a model of affordance suitable for use in multi-agent simulation is presented in a number of different ways.

First the properties of affordances used in the model and the aspects of the simulation that need to be considered in the model are presented. Second a description of affordances as relations between agents and other entities in the environment is described. Third, a sample specification of an affordance based agent reasoning model is presented. This model is specified using the Z formal specification language [140] and demonstrates the different stages of affordance based agent reasoning in the context of a basic OODA loop model. Fourth a number of illustrative examples from the game Capture The Flag are used to explain how the model of affordance can be used in a multi-agent simulation. Finally, the Chapter discusses how the model of affordance can be adopted for use in an actual multi-agent simulation, specifically at the algorithmic level.

5.2 Affordances in Ecological Psychology

It is important to emphasise that the motivation for this research is not driven by a need to create a high fidelity and cognitively realistic model or simulation of affordance. Rather, the theory of affordances serves as a guide and inspiration for engineering improved agent – environment interactions in multi-agent simulations.

However, ongoing research into the nature of affordances by the ecological psychology and situated cognition communities has the potential to inform the design of affordance based multi-agent simulations in a number of ways. First, it helps the simulation designer identify the important and relevant issues when modelling affordances. Second, it helps to determine what characteristics of the agent, environment and their relationship are important.

Due to the fact that there is an ongoing debate about the nature of affordances in the ecological psychology community [88, 142, 96, 181] and therefore a number of definitions, the concept of an affordance may appear nebulous to a software designer.

Gibson proposed the idea of affordances as part of an ecological approach to visual perception [53]. Affordances were described by Gibson as action possibilities or opportunities for action which humans and animals can perceive in the environment. This description strongly suggests a set of specific design requirements for agent – environment systems. This not only includes the design requirement that both agent percepts and actions are explicitly represented but also includes the need for agents to perceive possible actions or opportunities for action in the environment.

These design requirements give rise to some practical engineering issues. First, while many agent languages support the explicit representation of mental attitudes (such as beliefs, desires and intentions), many do not support an explicit representation of percepts and actions and allow for an arbitrary interface with the virtual environment.

Second, any agent – environment system [45] must allow for an agent to perceive action possibilities with respect to other entities and agents in the world. That is, agent perception in virtual environments needs to go beyond representing the ability of an agent to perceive simple properties of entities in the world.

Rather, models of perception need to be extended to take into account perception of possible actions that an agent can take with respect to an entity. This implies that action possibilities which some entity affords an agent will not necessarily be the same for all agents. This is because the possible actions that an agent can undertake in the world will be dependent on the exact capabilities of the agent. Gibson also separated the idea of affordance perception from object classification.

[45] In the real world the complex system consisting of the environment and the agents such as humans and animals which inhabit it, are often referred to as an ecology or an ecosystem. Multi-agent simulation can in some cases be considered the virtual or cyber analogues of these systems. This is especially the case when the multi-agent simulations are models of a real world ecology. A computational agent – environment system may then be considered as a virtual ecology.

"The fact that a stone is a missile does not imply that it cannot be other things as well. It can be a paperweight, a bookend, a hammer, or a pendulum bob. It can be piled on another rock to make a cairn or a stone wall. These affordances are all consistent with one another. The differences between them are not clear-cut, and the arbitrary names by which they are called do not count for perception. If you know what can be done with a graspable detached object, what it can be used for, you can call it whatever you please."

— James J. Gibson [54]

There are a number of other aspects of Gibson's description of affordances which need to be considered if they are to be used as a mechanism for agent – environment interaction. These are the direct perception of affordances, their meaning, their relational and their intentional nature. Gibson proposed that affordances can be directly perceived in the environment. That is, humans and animals do not only perceive the properties of objects in the world, but they can directly perceive the possible actions which those objects afford them.

This is in contrast to inferential theories of perception, where determining what action to take is a result of inferencing or other reasoning processes based on the perception of an object's properties in the world. However, most models of agent interaction and reasoning tend to be more closely aligned to inferential rather than direct theories of perception. This poses a challenge for modelling, designing and implementing affordance based interactions in multi-agent simulations.

The decision to adopt a direct perception model of affordance for a multi-agent simulation means that the design of the virtual environment and the agent – environment interaction needs to be considered in more detail. It means that the virtual environment must be able to model and represent the direct perception of action possibilities. On the other hand, if affordances are treated as mental attitudes, it makes it easier to integrate into existing models of agency, but this means that there is a greater deviation from Gibson's view of affordances.

The second aspect of Gibson's view of affordances that needs to be considered is that of meaning. Gibson originally meant for the concept of affordance to deal with the issue of meaning in psychology. Agents situated in a virtual environment should be able to perceive meaningful information about the world. This means that the affordances that each agent can perceive should be meaningful to them. For example, in the air combat domain, fighter pilot agents should perceive action possibilities that are meaningful, understandable and relevant to the current situation.

The third aspect of Gibson's view of affordances that needs to be considered is their inherently relational nature. In many multi-agent simulations, agents are designed to perceive and reason about entities in the world. However, as pointed out by Chemero [23], humans can perceive relations in the world and affordances are a special type of relation.

Chemero uses the example of one person being taller than another. For example, if Mary is taller than Anna (*taller(Mary, Anna)*) this concept is perceivable and therefore exists. However, the relation is not part of Mary or Anna but depends on both of them. Chemero argues that ffordances are similar. They do not exist in humans or in the environment, but come about because of their interaction. How does one then model and design a multi-agent architecture along these lines? It raises a number of practical, design and engineering issues which need to be considered.

Perhaps the best known use of the concept of affordance outside the ecological psychology community comes from the field of industrial design. In the book *The Design of Everyday Things*, Norman looked at affordances provided by everyday objects in the context of industrial design and user interfaces for various systems [104]. Norman's view of affordances varies from Gibson's:

> *"To Gibson, affordances are relationships. They exist naturally: they do not have to be visible, known or desirable".*
>
> — Donald Norman [103]

In attempt to describe how humans know what to do with an object when they have not seen it before, Norman said:

> *"The answer, I decided, was that the required information was in the world: the appearance of the device could provide the critical clues required for its proper operation."*
>
> — Donald Norman [103]

Norman also makes a distinction between perceived and real affordances. In industrial design what is important is what a user perceives to be possible with an object rather than what is really possible – which is closer to Gibson's view of affordances. The distinction potentially has some important ramifications for agent behaviour in multi-agent systems. One can imagine that the outcome of an air combat engagement could substantially change if a fighter pilot agent took a course of action based on a perceived affordance that was not real.

The debate in the ecological psychology community on the nature of affordances can give further insight into some of the potential practical issues that may be faced in modelling affordances in a virtual ecology. One question that has implications for the design of of multi-agent simulation is whether affordances exist independently of their perception by agents. It is generally agreed by ecological psychologists that affordances do not have to be perceived to exist.

> *"Affordances are not created in the act of perception; they exist independent of it. The theory of affordances is part of the ecological ontology. It is a statement of what is available in the world to be perceived."*
>
> — Claire F. Michaels [97]

While not the mainstream view in ecological psychology, some cognitive scientists consider affordances to correspond to mental attitudes regarding action possibilities. If this view is adopted, an affordance can simply be implemented as a mental state inside an existing agent. In a BDI based agent this may be as simple as representing the affordance as a belief that a particular action is possible with respect to some entity in the world.

While not consistent with the mainstream view in ecological psychology, adopting this type of affordance in a multi-agent simulation may be desirable in some circumstances, as it neatly fits into existing virtual environment architectures and agent reasoning engines. Adopting the mainstream view (in ecological psychology) of an affordance for a multi-agent simulation has a greater impact as the design of the environment, the agent and the agent – environment interaction all need to be considered.

The affordances that an agent can perceive in the virtual world should be directly related to the actions that agent can undertake. In fact it is strongly argued by Michaels [97] that affordances must be related to agent's actions. This means that how agents are embodied in a virtual environment is critically important in determining the affordances that they can perceive.

In an air combat simulation, the affordances perceivable by a pilot are determined and constrained by the pilot's embodiment as described in Chapter 2. This goes beyond modelling the human body and extends to the entire aircraft system (platform, weapon, sensors etc.). In this case the pilot is embodied in a fighter aircraft and the actions that the pilot can undertake are defined by the capabilities of the aircraft as an entire system. Agent embodiment (regardless of the body model used), is therefore a critical concept which needs to be addressed in a multi-agent simulation making use of affordance based interaction.

What must also be taken into account in an implementation of an affordance based multi-agent simulation is the aspects of the agent needed to determine what affordances are available. In addition to an agent's embodied action capabilities, an agent's mental state may need to be taken into account when determining affordances. One can argue that an agent's beliefs about itself and the world will strongly affect the type of affordances it can perceive. However, a number of ecological psychologists have emphasised the goal directed or intentional nature of affordances [128].

Affordances are not simply the list of all possible actions an agent can undertake in the world at any given time. Rather they are the relevant goal directed actions. In other words, affordances can be considered as the possible actions that an agent can undertake with respect to its own intentions. This means that in order to design and construct an affordance based virtual ecology, modelling an agent's intentions is important. How intentions manifest themselves in an affordance oriented agent will depend on the application domain at hand and the specific requirements for the multi-agent system being developed.

The other aspect of a multi-agent simulation which needs to be carefully considered is the design of the virtual environment. In his outline of a theory of affordances, Chemero [23] discusses the nature of the environment in the context of perception theories.

> *"In inferential theories of perception, these meanings arise inside animals based upon their interactions with the physical environment. In direct theories of perception, on the other hand, meaning is in the environment, and perception does not depend upon meaning-conferring inferences. Instead the animal simply gathers information from a meaning laden environment. But if the environment contains meanings, then it cannot be merely physical."*
>
> — Anthony Chemero [23]

This means that if affordances are to be modelled and represented in a multi-agent simulation the virtual environment must go beyond just representing the physical environment as discussed in Chapter 4. The term environment must be extended past the physical representation of the world and include a number of different modalities. Meaningful labels in terms of action possibilities or affordances that are relevant to individual agents are only part of what needs to be considered.

What is possible in the world for a particular agent depends on many different environmental modalities. Not only does it depend on the physical presence of entities in the world, but it also depends on the social environment. This is especially true in a military simulation where the environments being considered are many and complex. The action possibilities that can be perceived in the world by a military operator depend on many different environmental modalities, therefore an affordance based virtual ecology needs to reflect this.

5.3 The Affordance Model

Affordances are the perceived possible actions that an agent can undertake with respect to some entity in the environment. Affordances have a number of important properties which need to be considered in the development of any interaction model in the context of a multi-agent simulation. The virtual nature of the environments being considered provides significant flexibility in how these properties are interpreted and ultimately manifest themselves in computational models and subsequently software architectures.

5.3.1 Relevant Properties of Affordances

Affordances have many properties which can be taken into account. However, here the five properties of the affordance concept most relevant to multi-agent simulation are listed.

In Section 4.2 five properties of affordances were listed that were considered in this research. These properties are revisited here because they are important in determining the model of affordance presented in this Section. It is the aim of this Chapter to capture these properties of the affordance concept in a model for the following reasons.

1. They are the most salient properties of the affordance concept (as described in the review of the nature of affordances from the ecological psychology literature in Section 5.2.
2. Existing approaches to computationally modelling affordances in multi-agent systems have not captured all these properties.
3. They are properties that can capture the types of agent – environment interaction which are of interest in military multi-agent simulations.

To recap from Section 4.2, the properties of affordances that are considered in this research are:

(1) Affordances are Action Oriented: Affordances are ultimately about the possible actions that agents can take in the world.

(2) Affordances are Meaningful: Affordances are meaningful to the each agent observer. They confer a possibility for action that is tailored, custom made, accessible and understandable for the agent perceiving a particular affordance.

(3) Affordances are Relational: Affordances are inherently a relational concept. Unlike the mental attitudes of the BDI model, affordances cannot be internal agent constructs resulting purely from agent reasoning. Neither are they properties of the environment. Rather they come about from the relationship and interaction between an agent and some entity in the environment.

(4) Affordances are Intentional: Affordances are intentional in nature. That is, the affordances an agent perceives in the world will be determined by their current intentions. What an agent wants to do in the world will determine how the agent views the world and hence determine what actions are possible.

(5) Affordances are Directly Perceivable: Affordances are directly perceived in the environment. In fact, the theory of affordances is considered the ontology which supports the concept of direct perception[46]. The idea that affordances can be directly perceived in the environment also re-enforces their relational nature.

These properties of affordances imply that agents will perceive different action possibilities with respect to the same entity in the world. Affordances are not properties of the agent, not properties of a perceived entity but depend on both and come about through the interplay of the two. This also implies that affordances are highly dynamic in nature. That is, the possible actions that an agent can undertake in the world is always changing.

5.3.2 Model Inputs

In order to determine the affordances a particular agent can compute with respect to some entity in the virtual environment in a multi-agent simulation, properties of the agent, the entity, the environment and their relationships all need to be considered. The designer of a multi-agent simulation should consider these attributes when designing a system which uses an affordance based approach.

Agent Mental State The affordances perceived by an agent are influenced by an agent's mental state. For example, if the BDI model is used to represent agent mental state, the affordances an agent perceives will be influenced by the agent's beliefs, desires and intentions. The intentional nature of affordance has been described in ecological psychology. Agents should perceive action possibilities relating to their current goals and intentions. Although some action may be possible for the agent to perform, it is not considered an affordance if it is not related to an agent's intentions. Similarly, an agent's beliefs can also influence the perceived affordances.

[46]Chemero [23] states — *"Thus like earlier theories that take perception to be direct, James Gibsons's ecological psychology includes an ontology, his theory of affordances."*

Agent Mental Capability An agent's mental capability refers to the type of goals that the agent can try to achieve.[47] Mental capability is important in determining affordances because an agent can perceive an affordance to perform some complex rather than an atomic exogenous action. Such complex series of actions are typically implemented as goals that an agent can achieve. Therefore, the possible affordances an agent can perceive are directly constrained by the types of goals the agent can undertake.

Agent Physical State An agent's physical state (whether it be size, speed, direction etc.) influence the types of affordances an agent can perceive. For example, some affordances may only manifest when the agent is in motion, or is in a particular orientation. The concept of agent embodiment is therefore fundamental in the concept of affordance.

Agent Physical Capability Ultimately the type of actions an agent can take in the world is constrained by the type of exogenous actions the agent can undertake in the environment. This is referred to as the agent's physical capability and similar to the agent's physical state is determined by the nature of the agent's embodiment.

Entity State The affordances with respect to some entity in the environment cannot be determined without some knowledge of the actual entity being considered. Therefore, knowledge of the state of the entity (properties, attributes and characteristics) is required.

Environment The possible affordances that can be perceived by the agent with respect to an entity will be influenced by the environment in which they are situated. Here the term environment is used in its broadest possible sense and includes the physical aspects of the environment (such as temperature, terrain), relational concepts (relative distance and orientation), and aspects of the social environment (such as team and organisational structures).

5.3.3 Procedural Model Description

Using this information a model of agent – environment interaction based around the notion of affordance perception was developed. The model can be described in a number of different ways. From a procedural perspective the steps involved in perceiving, reasoning and acting in an affordance based multi-agent simulation can be described in this model as follows.

1. Determine which other entities (including agents) each agent in the simulation can perceive. This is determined by the suite of sensors available to each agent.
2. For each agent-entity pair determine what affordances arise from this particular interaction.
3. The agent is then presented with a list of affordances that are available to it at that point in time. For each entity the agent can perceive there may be zero, one or more affordances available.

[47]Typically in BDI programming languages this is implemented as a plan library.

4. The agent must then decide which (if any) of these affordances it wishes to adopt.
5. Once a decision has be made to adopt an affordance (a possible action), the action must then be enacted in the world. This then takes the action from possibility to manifestation in the environment.

Step (ii) attempts to determine the affordances available for each agent-entity interaction in this simulation. This determination takes into account the perceivable properties of the entity, the agent's mental and physical state, the agent's physical and mental capabilities as well as environmental relational properties arising from the interaction. This list of affordances (or possible actions that the agent can undertake in the world) that are determined by this process are constrained by the agent's physical and mental capabilities. Therefore the list of affordances available for each agent can only come from the list of things the agent is capable of doing. In other words, affordances do not include actions that an agent can't undertake with respect to an entity in the world.

Overall this model can be considered to have two important steps. The first is determining which affordances are available and the second is deciding which affordance to adopt. If the steps to perceive and act in the environment are added, the resulting four step reasoning model is conceptually very similar to high level fighter pilot reasoning models based on Boyd's OODA (Observe-Orient-Decide-Act) loop that have been used in air combat simulations. This means that an affordance based agent reasoning model can be framed in terms of the OODA based agent cognitive models as follows:

- **Observation:** Observe The World
- **Orientation:** Find Affordances
- **Decision:** What Affordances to Adopt
- **Action:** Perform Action of Adopted Affordance

5.4 Affordances as Relations

Affordances are typically described as not being part of the agent or the environment but rather come about due to the interaction between the two. This Section deals with this relational property of affordances. The approach taken starts by looking at the affordances an agent can perceive with respect to a single entity in the environment. These affordances are represented as relationships between the agent and the entity in the environment.

5.4.1 Definitions

Consider a multi-agent system S in which the environment consists of a number of entities, some of which are agents. Traditionally agents are differentiated from regular entities or objects in multi-agent systems by the agent's abilities to perceive and act in an environment, as opposed to regular objects which are affected by the environment or other entities. In the system being considered, agents will be further differentiated from objects by an agent's ability to perceive affordances or action possibilities with respect to other entities (whether they be other agents or objects) in the environment.

Assume that the environment consists of a set of entities E. The subset of entities which are agents is given by the set A, and hence $A \subseteq E$. In the case where all the entities are agents such that $A = E$ we have a pure multi-agent system. The number of agents and the total number of entities in these multi-agent simulations can be considered to be finite. It is possible for an affordance relationship to exist between an agent i and an entity j, where $i \in A$ and $j \in E$. This implies that entity j affords agent i some possible action at a particular time t. This can be defined as an affordance relation ϕ as shown in Definition 3.

Definition 3 (Affordance Relation – Single Affordance)**.**

$$\phi_{ij}(t) = \phi(i, j, t) \tag{2}$$

It is possible for an agent i to have more than one affordance with respect to entity j. We use k to denote the specific affordance relation between i and j. The notation for the affordance relation is shown in Definition 4.

Definition 4 (Affordance Relation – Multiple Affordances)**.**

$$\phi^k_{ij}(t) = \phi(i, j, k, t) \tag{3}$$

An alternative form for this affordance relationship is also possible using the functional form $\mathsf{affords}$ as shown in Definition 5.

Definition 5 (Affordance Relation - Alternative Form)**.**

$$\phi^k_{ij}(t) = \mathsf{affords}(i, j, \phi^k) \tag{4}$$

The set of all possible affordances between agent i and entity j is at time t can then be denoted as $\Phi_{ij}(t)$. When K is the set of all possible affordances entity j affords agent i then $\Phi_{ij}(t)$ can be defined as shown in Definition 6.

Definition 6 (All Affordances between Agent i and Entity j)**.**

$$\Phi_{ij}(t) = \bigcup_{k \in K} \phi^k_{ij}(t) \tag{5}$$

where K is the set of all possible affordances j affords i.

It is important to note that two different agents α and β will not necessarily have the same set of affordances with respect to an entity γ. It is not necessarily the case that $\Phi_{\alpha\gamma} = \Phi_{\beta\gamma}$. Similarly, the set of affordances that α has with respect to β are not necessarily the same as those that β has with respect to α, and therefore in most cases $\Phi_{\alpha\beta} \neq \Phi_{\beta\alpha}$.

The set of all entities in the world excluding entity (agent) i is defined as the set $E'(i)$.

Definition 7 (Entities other than Agent i)**.**

$$E'(i) == \{\forall j : E \mid i \neq j \bullet j\} \tag{6}$$

The action possibilities afforded to an agent i at time t in a multi-agent system is given by the union of all affordances between agent i and entity j, excluding affordances with respect to itself (that is the case where $i = j$ is not considered):

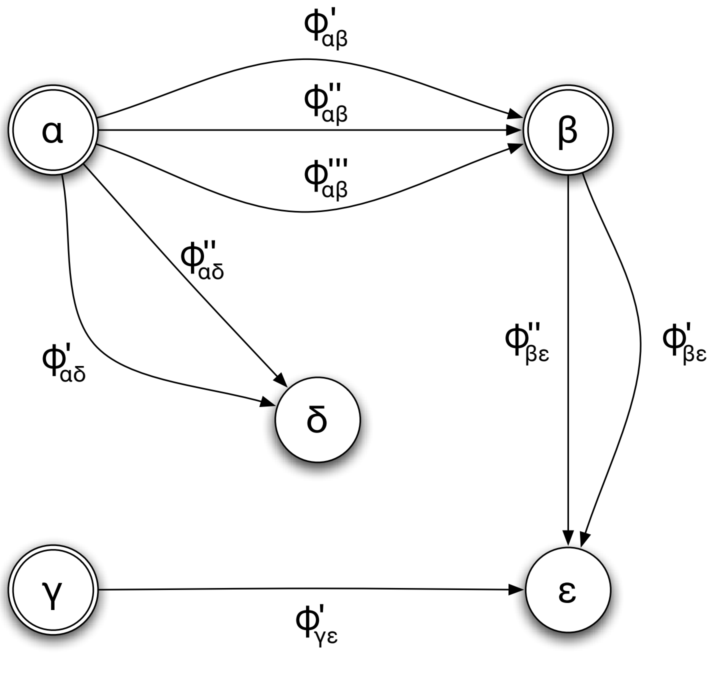
Definition 8 (Affordances for Agent i)**.**

$$\Phi_i(t) = \bigcup_{j \in E'(i)} \Phi_{ij}(t) = \bigcup_{j \in E'(i)} \bigcup_{k \in K} \phi_{ij}^k(t) \tag{7}$$

The complete set of affordances in the multi-agent system at any given time t is then given by the set of affordances for every agent in the system:

Definition 9 (All Affordances for Multi-Agent Simulation)**.**

$$\mathbf{\Phi_{MAS}}(t) = \bigcup_{i \in A} \Phi_i(t) = \bigcup_{i \in A} \bigcup_{j \in E'(i)} \bigcup_{k \in K} \phi_{ij}^k(t) \tag{8}$$



Figures 5.1: An Example of an Affordance Relationship Graph for multi-agent system S. The nodes in the graph represent entities in the simulation at time t. Some of the entities are agents and are denoted by the double circled nodes. In this system, the agents are α, β and γ. Affordances are represented as directed arc between an agent and another entity (which may also be an agent). Each k^{th} affordance relationship between agent i and entity j is denoted by the symbol ϕ_{ij}^k as a label on a directed arc on the graph.

5.4.2 Graph Representation

It is possible to consider the affordance relations between agents and other entities in the world using a graph representation with the following properties:

1. The entities in the world are represented by the graph nodes.
2. There are two types of graph nodes, one for agents and the other for non-agent entities.
3. An arc between two nodes represents an affordance.
4. All affordance arcs are directional. They can only emanate from agents, but can terminate at any other entity. This indicates that only agents can perceive affordances with respect to other entities in the world.
5. Each arc is uniquely labelled and represents a unique affordance or action possibility.
6. It is possible for multiple arcs to emanate from the same node and converge to another node, representing multiple affordances.
7. Since affordances are dynamic, the graph is also dynamic and the arcs change with time.

Consider a multi-agent system S represented as a graph. The multi-agent system consists of a number of entities E, a subset of which A, are agents, such that:

$$E = \{\alpha, \beta, \gamma, \delta, \epsilon, \zeta\} \tag{9}$$

$$A = \{\alpha, \beta, \gamma\} \tag{10}$$

Since agents do not consider affordances with respect to themselves the set $E'(i)$ for each agent in simulation S is as follows.

$$\begin{array}{rcl} E'(\alpha) & = & \{\beta, \gamma, \delta, \epsilon, \zeta\} \\ E'(\beta) & = & \{\alpha, \gamma, \delta, \epsilon, \zeta\} \\ E'(\gamma) & = & \{\alpha, \beta, \delta, \epsilon, \zeta\} \end{array} \tag{11}$$

At a particular time t, the affordances for each agent in the set A are given by (omitting the time dependence for conciseness):

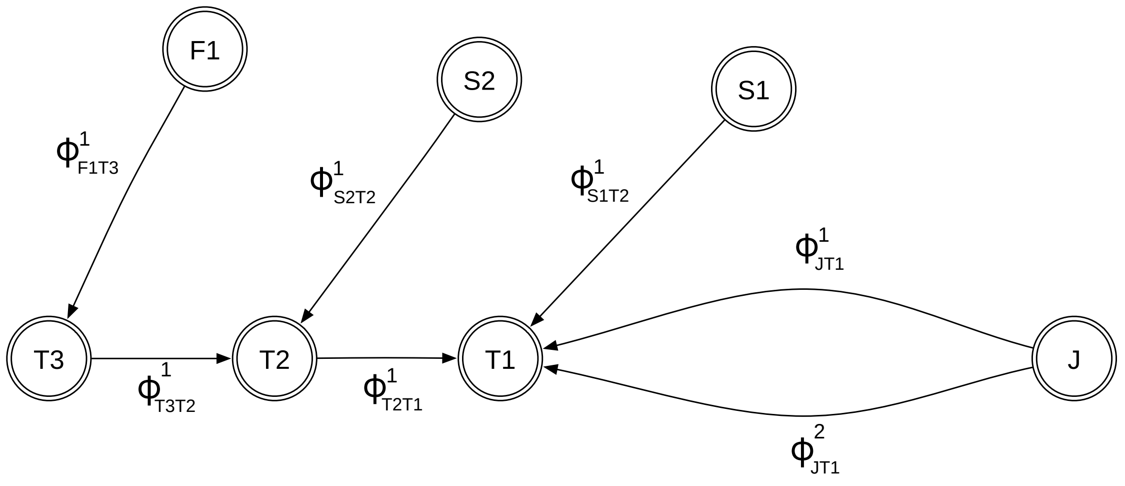
$$\begin{array}{llll} \Phi_{\alpha\beta} = \{\phi'_{\alpha\beta}, \phi''_{\alpha\beta}, \phi'''_{\alpha\beta}\} & \Phi_{\alpha\gamma} = \{\} & \Phi_{\alpha\delta} = \{\phi'_{\alpha\delta}, \phi''_{\alpha\delta}\} & \Phi_{\alpha\epsilon} = \{\} \\ \Phi_{\beta\alpha} = \{\} & \Phi_{\beta\gamma} = \{\} & \Phi_{\beta\delta} = \{\} & \Phi_{\beta\epsilon} = \{\phi'_{\beta\epsilon}, \phi''_{\beta\epsilon}\} \\ \Phi_{\gamma\alpha} = \{\} & \Phi_{\gamma\beta} = \{\} & \Phi_{\gamma\delta} = \{\} & \Phi_{\gamma\epsilon} = \{\phi'_{\gamma\epsilon}\} \end{array} \tag{12}$$

The affordances for the three agents α, β and γ in the system can then be written as follows (omitting any affordances that do not exist):

$$\begin{array}{rcl} \Phi_\alpha(t) & = & \{\Phi_{\alpha\beta}(t), \Phi_{\alpha\delta}(t)\} \\ \Phi_\beta(t) & = & \{\Phi_{\beta\epsilon}(t)\} \\ \Phi_\gamma(t) & = & \{\Phi_{\gamma\epsilon}(t)\} \end{array} \tag{13}$$

Therefore, the complete set of affordances in the multi-agent system S represented as a graph can be defined as:

$$\mathbf{\Phi_S(t)} = \{\Phi_\alpha(t), \Phi_\beta(t), \Phi_\gamma(t)\} \tag{14}$$



Figures 5.2: An example affordance graph depicting affordance relationships between agents from the domain of Close Air Support (CAS) from the Human Agent Virtual Environment (HAVE) as described in Chapter 2. The nodes in this graph represent different agents in the HAVE multi-agent simulation at a time t. Since all the nodes in this case represent agents, they are depicted using a double circle. In this graph agents $T1$, $T2$, and $T3$ represent enemy tanks in a convoy. J represents an agent undertaking the Joint Terminal Air Controller (JTAC) role. $F1$ represents a fighter aircraft on combat air patrol, while $S1$ and $S2$ represent strike fighter aircraft on a close air support mission. Affordance relationships between these agents are represented by the labelled arcs and are described in further detail in the text.

5.4.3 Illustrative Example of Affordance Graph

The graph representation in the previous section detailed an abstract example. Here a more concrete example is adapted from the domain of Close Air Support from the HAVE application described in Chapter 2. A graph representation this scenario is depicted in Figure 5.2 and is described further in this Section.

The scenario being considered is similar (although not identical) to the one depicted in Figure 2.3. In this scenario in the multi-agent simulation *HAVE* at time t the following agents are present.

Striker $S1$ and $S2$: The strike fighters $S1$ and $S2$ are tasked with destroying enemy tanks. The must positively identify an enemy tank before it can be destroyed.

Fighter $F1$: The fighter aircraft $F1$ flies high above the battle field in a combat air patrol mission. Its role is to defend the air space against other enemy aircraft and hence is not directly concerned with the situation on the ground.

JTAC J: The Joint Terminal Air Controller (JTAC) denoted by $J1$ is on the ground with troops and can report the position of enemy tanks and call in air strikes against them.

Tanks $T1$, $T2$, $T3$: Tanks $T1$, $T2$ and $T3$ represent a convoy of enemy tanks.

In this scenarios all the entities in this simulation also happen to be agents. Therefore we can define the set of entities E and agents A as follows.

$$E = \{S1, S2, F1, J, T1, T2, T3\}$$
$$A = \{S1, S2, F1, J, T1, T2, T3\}$$

The affordances which each agent in the simulation perceives at time t can now be considered. The JTAC J can perceive two affordances with respect to the lead tank $T1$ in the convoy. The tank $T1$ affords having its position being reported or having a strike being called against it. These relationships can be represented as:

$$\begin{array}{rcl} \phi^1_{JT1}(t) & = & \mathit{ReportPosition} \\ \phi^2_{JT1}(t) & = & \mathit{CallInStrikeAgainst} \end{array} \tag{15}$$

Therefore the complete set of affordance relations which the JTAC can perceive [48] can be denoted as:

$$\Phi_J(t) = \{\phi^1_{JT1}, \phi^2_{JT2}\} \tag{16}$$

Since the tanks are in a convoy, tank $T1$ affords being followed by tank $T2$ and similarly tank $T2$ affords being followed by tank $T3$. These affordance relationships can be represented as follows:

$$\begin{array}{rcl} \phi^1_{T2T1}(t) & = & \mathit{Follow} \\ \phi^1_{T3T1}(t) & = & \mathit{Follow} \end{array} \tag{17}$$

If it is assumed that in the current time step t that tank $T1$ cannot perceive any affordances, then the complete set of affordances for each of the three tanks can be denoted as follows:

$$\begin{array}{rcl} \Phi_{T1}(t) & = & \{\} \\ \Phi_{T2}(t) & = & \{\phi^1_{T2T1}(t)\} \\ \Phi_{T3}(t) & = & \{\phi^1_{T3T2}(t)\} \end{array} \tag{18}$$

At time t the fighter aircraft $F1$ can only perceive tank $T3$ on the ground due to its position in the air. Since its mission is to patrol the airspace, agent $F1$ is not directly interested in tank $T3$ and hence it affords ignoring.

$$\phi^1_{F1T3}(t) = \{\mathit{Ignore}\} \tag{19}$$

Hence the complete set of affordances perceived by agent $F1$ is denoted by:

$$\Phi_{F1}(t) = \{\phi^1_{F1T3}(t)\} \tag{20}$$

In the case of the two strike fighter aircraft $S1$ and $S2$ at time t, $S1$ has identified the lead tank $T1$ which affords being attacked, while $S2$ has yet to identify tank $T2$, hence this tank affords being identified.

$$\begin{array}{rcl} \phi^1_{S1T1}(t) & = & \mathit{Attack} \\ \phi^1_{S2T2}(t) & = & \mathit{Identify} \end{array} \tag{21}$$

[48] Note that in this case the JTAC happens to only be perceiving affordances with respect to one entity in the environment, in this case the lead tank $T1$.

Therefore the set of affordances for strike fighter agents $S1$ and $S2$ can be denoted as follows:

$$\begin{array}{rcl} \Phi_{S1}(t) & = & \{\phi^{1}_{S1T1}(t)\} \\ \Phi_{S2}(t) & = & \{\phi^{1}_{S2T2}(t)\} \end{array} \tag{22}$$

Therefore, the complete set of affordance relationships at time t in the *HAVE* multi-agent simulation can be denoted as follows:

$$\mathbf{\Phi_{HAVE}}(t) = \{\Phi_{J(t)}, \Phi_{T1}(t), \Phi_{T2}(t), \Phi_{T3}(t), \Phi_{F1}(t), \Phi_{S1}(t), \Phi_{S2}(t)\} \tag{23}$$

As mentioned previously a graphical representation of these relationships is shown in Figure 5.2. Further illustrative examples from the game Capture The Flag can be found in Section 5.8.

5.5 Algorithm for Finding Affordance Relations

The generation of the affordance relations between agents and entities in a multi-agent simulation as described in Section 5.4 can be captured in the form of an algorithm.

Algorithm 5.1 provides an example of how the relational model can be adapted to an algorithm suitable for inclusion in an environment component or a component that acts as the interface between the agents and the environment in a multi-agent simulation. The four inputs to the algorithm are

- the *agents* in the multi-agent system,
- all the *entities* in the system (which includes all the agents),
- a list of all the affordances that will be considered Φ and
- the current simulation time t.

The list of affordances Φ being considered is the list of all possible actions that can be taken in the world.[49] Taking these parameters into account, the algorithm produces the set of affordances for each agent-entity relation at time t which is denoted by the set affordances. This corresponds to the complete set of affordances in a multi-agent system at time t as denoted by $\mathbf{\Phi_{MAS}}(t)$ in Equation 8.

The algorithm considers the relation between each agent and every other entity in the multi-agent simulation. For each such agent-entity relation, the algorithm then considers every possible affordance that could be provided by the entity to the agent. If the entity affords some possible action to the agent, then it is added to the list of affordances for that particular time. The algorithm then returns a set of all the available affordances in the multi-agent simulation at the specified time.

These affordances can then be used by a multi-agent architecture in one of two ways.

[49] In many cases it is expected that the list of all possible actions will be pre-determined by the designer of the multi-agent simulation in question.

Algorithm 5.1 Finding All Affordance Relations in a Multi-Agent Simulation

```
function FindAffordanceRelations($agents$, $entities$, $\Phi$, $t$)
    affordances($\alpha, \varepsilon, t$) $\leftarrow \{\}$
    for all $\alpha \in agents$ do
        for all $\varepsilon \in entities$ do
            if $\varepsilon \neq \alpha$ then
                for all $\phi \in \Phi$ do
                    if affords($\alpha, \varepsilon, \phi$) $= true$ then
                        affordances($\alpha, \varepsilon, t$) $\leftarrow \phi$
                    end if
                end for
            end if
        end for
    end for
    return affordances
end function
```

- The generated affordances can be added to the representation of the virtual environment as annotations allowing for agents to directly perceive them.
- The generated affordances can be sent directly to each agent, allowing each agent to be *told* what affordances are available to them.

In both these cases the affordances are not computed directly by each agent. The agent however still needs to determine which, if any, affordances to adopt and to subsequently undertake the appropriate actions in the world.

This approach for computing affordances has a number of advantages.

1. It adheres more closely to the view of affordances from ecological psychology in that it captures the concept of affordance as a relation and also allows for representing the direct perception of affordances in the environment.
2. It also captures the view that affordances can exist independent of their perception by an agent.
3. It allows for the affordances to be computed at run time or to be pre-computed and simply evaluated by the algorithm to see if it exists.
4. It off loads some of the intelligence related computation that would traditionally be undertaken in the agent and places it in the environment. This has the potential to result in simpler and less complex designs and implementations of individual agents. It also means that behaviour of agents in an environment can be changed without making changes to an agent.
5. It relates an agent's percepts to an agent's actions.

5.6 Affordance Based Agent Reasoning

Section 5.4 described a high level and abstract model of affordance in terms of a relation between an agent and some other entity in the world which affords an agent some course of action. In this Section the nature of the affordance relation is represented at its most basic level; the relationship between a single agent and a single entity in the environment.

In order to design and develop useful and working simulation systems, the nature of the relation $\phi^k_{ij}(t)$ needs to be expanded and articulated further. In order to do this, the various properties of affordances described in Section 5.3.1 need to be taken into account.

This Section presents an agent reasoning model that is affordance based. That is it looks at the generation of affordances from the perspective of each individual agent, rather than the global, environmental and relational perspective presented in previous Sections.

The model is defined and presented using the formal specification language Z [140, 175]. The Z specification language makes use of set theory together with predicate and propositional logic to formally specify the requirements of the software system. It has been used here because Z allowed the affordance based model to be specified in an abstract mathematical manner independent of any particular implementation language or approach. It was also chosen because Z has been used a number of times in the multi-agent systems community to specify models of agency, such as its use to formally specify the dMARS BDI agent programming language [35, 33] as well as the specification of the SMART agent framework [34].

Boyd's OODA (Observe-Orient-Decide-Act) loop is used a mechanism for structuring and organising the various reasoning steps in the model. This model also follows the traditional agent design pattern of *Perception-Reasoning-Action.* That is, the agent perceives the environment through various sensors, undertakes some reasoning and then undertakes actions in the environment.

In this model, percepts and actions are related in the agent reasoning process through affordances. Specifically what is of interest is the affordances the agent can perceive with respect to particular entities in the environment. Therefore, the agent must have knowledge and be able to distinguish between different entities in the environment.

5.6.1 Definitions

The set of all entities is introduced.

$[Entity]$

An agent needs to be able to perceive the entities in the world, so an abstract introduction of the set of percepts is also required.

$[Percept]$

Without defining the exact nature of an agent's intentions, the set *Intention* is introduced. This is not only because affordances are intentional in nature, but also because of the importance intentions play in agent models such as the BDI model.

$[\mathit{Intention}]$

In cognitive science, affordances are often explained in term of action possibilities that a human or animal can undertake in the real world. In a virtual environment, one might also consider affordances as possible actions an agent can undertake with respect to entities.

The set of possible actions an agent can undertake in the world is constrained by the agent's embodiment or physical capabilities. These can be considered as atomic exogenous actions that the agent undertakes in the environment (as opposed to internal mental actions), and are introduced as follows.

$[\mathit{PossibleAction}]$

However, if affordances (or action possibilities) were constrained to simple exogenous actions, then there would be limitations in using such an approach for developing complex agent behaviour. If the concept of affordance for agents is extended to not only an action possibility, but to allow for the possibility of a complex series of actions with respect to a particular entity in the environment, then agents can perceive possible courses of action in the world with respect to particular entities.

The concept of a possible intention an agent can have is introduced and in this example is simply defined as a sequence of possible actions. It is referred to as a possible intention because it does not become an actual intention until the agent adopts it.

Also, the structure of the possible intention is deliberately kept simple (as a sequence of possible actions).[50] For the purposes of this example however, representing a possible intention as a sequence of possible actions is sufficient.

$\mathit{PossibleIntention} : \mathrm{seq}\, \mathit{PossibleAction}$

An affordance is often described in ecological psychology as an action possibility an agent (typically a human or animal) perceives with respect to some entity in the environment. The concept of an affordance is inherently a relational one; the relationship between an agent and an entity is defined through the possible actions that the agent can perceive with respect to the entity.

The idea of an affordance also goes beyond the concept of the possibility or opportunity for simple exogenous agent action. It also includes the possibility of an agent perceiving opportunities to adopt complex actions; that is for the agent to be able to perceive intentional possibilities. Therefore, in this model of agent reasoning, for a given agent, the concept of an affordance can be defined as a relation between a specific entity and a possible intention the agent can undertake with respect to that entity.

$\mathit{Affordance} : \mathit{Entity} \nrightarrow \mathit{PossibleIntention}$

With respect to a given agent, this defines an affordances as a relationship between an entity the agent can perceive and a possible intention the agent can have with respect to that entity.

[50]This concept is often implemented in BDI based languages as a plan typically caters for more complex structures such as branching and sub-goals.

It is important to note that the idea of a possible intention an agent can have may not at once seem to be consistent with the notion of an affordance from ecological psychology. However, if the only types of action possibilities that were considered were related to simple exogenous actions, then the concept of an affordance oriented mechanism for agent – environment interaction would have limited value. It is the complex series of actions or intentions that an agent can possibly have in the environment that makes the prospect of affordance oriented interaction in multi-agent simulations an interesting one.

5.6.2 A Model of an Affordance Based Agent

These definitions allow for the introduction of the *Agent* schema. In this model the agent has beliefs about a set of entities that it can perceive in the world. The agent also needs to know about the affordances it can perceive in the world as well as its set of current intentions.

The agent's capabilities are defined by its mental capability which is simply the set of all possible intentions that this agent can have, as well as the agent's physical capability which is defined by the exogenous actions the agent can undertake in the environment. The agent cannot take any actions in the environment, perceive the environment, or have any situated presence without being embodied. Hence the agent's body is included in the agent definition.

Agent

$$\begin{array}{l}
\mathit{entities} : \mathbb{P}\,\mathit{Entity} \\
\mathit{affordances} : \mathbb{P}\,\mathit{Affordance} \\
\mathit{intentions} : \mathbb{P}\,\mathit{Intention} \\
\mathit{mentalCap} : \mathbb{P}\,\mathit{PossibleIntention} \\
\mathit{physicalCap} : \mathbb{P}\,\mathit{PossibleAction} \\
\mathit{body} : \mathit{Body}
\end{array}$$

The agent reasoning model is split into four separate steps approximately following the *Observe-Orient-Decide-Act* or OODA Loop model of decision making.

5.6.3 Observing Entities in the Environment

The *Observe* schema receives a set of new percepts as inputs, updating the information about entities it can perceive using the *UpdateEntities* function.

Observe

$$\begin{array}{l}
\Delta \mathit{Agent} \\
\mathit{percepts?} : \mathbb{P}\,\mathit{Percept} \\
\mathit{UpdateEntities} : (\mathbb{P}\,\mathit{Percept} \rightarrow \mathbb{P}\,\mathit{Entity}) \rightarrow \mathbb{P}\,\mathit{Entity} \\
\hline
\mathit{entities}' = \mathit{UpdateEntities}(\mathit{percepts?}, \mathit{entities})
\end{array}$$

- The statement Δ*Agent* denotes that the *Observe* schema is going to make a change to the *Agent* schema. This introduces two sets of variables in the *Observe* schema.

First, it introduces the complete set of variables from the *Agent* schema that denote the state of the agent prior the *Observe* operation being undertaken. Second, it introduces a primed set of variables (for example $entities'$ denoting the state of the agent schema after the *Observe* operation has been completed).

- The single input to the observation step is a set of percepts denoted by the variable $percept?$. [51]

- A function *UpdateEntities* is defined that takes two parameters, a set of percepts and a set of entities. The function returns a set of entities. This function represents the updates that an agent makes about the knowledge of entities in the world that it has, based on new perceptual inputs.

- The final statement in the schema indicates that the agent's knowledge about the entities in the world denoted by $entities'$ has been updated based on knowledge about the previous set of entities and new percepts.

5.6.4 Orienting and Perceiving Affordances

In the *Orient* schema, the agent orients itself in the world determining what opportunities for action are available to it, based on the set of currently perceived entities. The agent does this by considering all possible entities that it can perceive. For each entity perceived, the agent considers what each entity affords the agent. In other words, the agent tries to find all possible affordances (or opportunities for action) with respect to the given entity.

The search for these affordances is left to the *FindAffordances* function which is deliberately left undefined because the exact definition will depend on the specific application domain in which the multi-agent simulation is being used. In order to determine the affordances for a particular entity the *FindAffordances* function not only considers the properties of the entity itself, but also considers the agent's intentions as well as its mental and physical capabilities.

Orient

$$\begin{array}{l}
\Delta Agent \\
FindAffordances : (Entity \rightarrow \mathbb{P}\, Intention \rightarrow \\
\qquad\qquad \mathbb{P}\, PossibleIntention \rightarrow \mathbb{P}\, PossibleAction) \\
\qquad\qquad\quad \rightarrow \mathbb{P}\, Affordance \\
\hline
\forall\, \varepsilon : Entity \mid \varepsilon \in entities \bullet \\
\quad affordances'(\varepsilon) = FindAffordances(\varepsilon, intentions, \\
\qquad\qquad\qquad mentalCap, physicalCap)
\end{array}$$

- The *Orient* schema includes the statement $\Delta Agent$ because it makes changes to the *Agent* schema.

[51]It is convention in Z specifications to denote schema input variables with a question mark (?) and schema output variables with an exclamation mark (!).

- The function *FindAffordances* uses a number of parameters to find the affordances with respect to a particular entity in the world. These parameters are the knowledge about the entity in question, the agent's current set of intentions as well as information about the agent's mental and physical capabilities.
- The final statement in the *Orient* schema indicates the change in the set of affordances the agent associates with each entity it can perceive in the world. For all entities in the set *entities*, the *FindAffordances* function finds all affordances associated with that entity taking into account information about the specific entity ε and information about its own intentions, mental and physical capabilities. The changes to the set of affordances the agent has after the *Orient* operation is denoted by the set $\mathit{affordances}'(\varepsilon)$ for each entity ε.[52]

5.6.5 Deciding on Affordance Adoption

Once the agent has determined what affordances are available with respect to every entity it can perceive it must then decide which one to adopt. This process is shown in the *Decide* schema. For each affordance that the agent knows about, the agent considers whether it should adopt it, using the *Adopt* function. If the agent decides to adopt a particular affordance it forms a corresponding intention and adds it to the existing set of intentions it is currently pursuing.

```
Decide
  $\Delta \mathit{Agent}$
  $\mathit{Adopt} : \mathit{Affordance} \rightarrow \mathbb{B}$
  $\mathit{FormIntention} : \mathit{Affordance} \rightarrow \mathit{Intention}$
  ----------------------------------------
  $\forall \phi : \mathit{Affordance} \mid \phi \in \mathit{affordances} \bullet$
      $\mathit{Adopt}(\phi) = \mathrm{true} \Rightarrow$
          $\mathit{intentions}' = \mathit{intentions} \cup \{\mathit{FormIntention}(\phi)\}$
```

- The Δ*Agent* statement denotes that the *Decide* schema will be changing the variables in the *Agent* schema.
- The function *Adopt* takes a single parameter, a specific affordance and determines if the agent should adopt that particular affordance. The implementation of the *Adopt* function will be dependant on the specific system being developed. The nature of this function is directly related to the concept of intention reconsideration. It is in this function that the agent considers its current intentions, the affordance in question and whether it should adopt it.
- The *FormIntention* function takes an affordance and forms an intention for the agent.
- The final statement in the schema indicates that for each affordances ϕ available to this agent it considers whether it should adopt them. If the agent decides to adopt

[52]How the *FindAffordances* function is actually implemented in a multi-agent simulation will vary from system to system. The set of design solutions is quite broad and may range from a complex computational process to a simple look-up of an affordance annotation in the virtual environment.

the affordance, then it forms and intention from the affordance and the resulting intention is added to the set of intentions the agent currently has. The agent's new set of intentions is denoted by the variable *intentions′*.

5.6.6 Acting in the Environment

The final step in the agent reasoning model is defined in the *Act* schema. It results in exogenous actions being produced as output from the schema. This is accomplished using the *Execute* function which executes the agent's current intentions resulting in new actions to be undertaken in the environment.

$$\begin{array}{|l}
\multicolumn{1}{l}{\textit{Act}} \\
\hline
\Delta \textit{Agent} \\
\textit{actions}! : \mathbb{P}\,\textit{PossibleAction} \\
\textit{Execute} : \mathbb{P}\,\textit{Intention} \rightarrow \mathbb{P}\,\textit{PossibleAction} \\
\hline
\textit{actions}! = \textit{Execute}(\textit{intentions}) \\
\hline
\end{array}$$

- The Δ*Agent* indicates that the *Act* schema will make a change to the agent.
- The variable *actions*! represents the exogenous actions that the agent will undertake in the world. They are from the set of possible actions that the agent can undertake.
- The *Execute* function takes the current intentions and transforms them into actions. How this is implemented will again depend on the particular system being considered.[53]
- The execution of the agent's intentions produces the only output to the schema denoted by *actions*!.

5.6.7 Affordance Oriented Reasoning

Having defined the *Observe*, *Orient*, *Decide* and *Act* schemas, these can be united into one schema called *Reason* which represents the reasoning process of an agent motivated in design by the theory of affordance.

The resulting *Reason* schema takes a set of percepts as input denoted by the *percepts*? variable in the *Observe* schema and produces a set of actions denoted by *actions*! in the *Act* schema. Hence, this model of affordance based agent reasoning is consistent with the traditional *Perceive-Reason-Act* agent model.

$$\textit{Reason} \mathrel{\widehat{=}} \textit{Observe} \wedge \textit{Orient} \wedge \textit{Decide} \wedge \textit{Act}$$

There are a number of interesting observations one could make about this agent reasoning model specification. First, the exact nature of the *FindAffordances* function is not specified.

[53]For example in a plan based BDI agent programming language this would correspond to the execution of a plan (representing a running intention).

Any function that determines the opportunities for action or affordances for an agent at any given time can be potentially computationally expensive. However, the computational complexity of a software implementation of *FindAffordance* will vary depending on the actual application domain.

Although the agent reasoning specification allows for the dynamic computation of affordances for each agent (i.e. at simulation run time), it does not preclude pre-computation, which allows for the *FindAffordance* function to be implemented as pre-defined database or simple table look-up. Second, the affordances for each agent are computed internally. This is at odds with current thought in ecological psychology where affordances are directly perceived by the agent and come about through an interacting relationship between agent and environment.

The question of whether affordances are computed in the agent or in the environment of a simulation system will be largely requirements driven. In this example it was assumed that the computation of affordances for each agent would occur inside the agent.

5.7 Relationship with BDI Agent Reasoning Models

The affordance based agent reasoning model presented above can be compared to more traditional agent reasoning models such as the Belief, Desires and Intentions (BDI) model. The idea of an affordance based reasoning model is not incompatible with the ideas from the BDI model. For example, one possibility is to implement affordances within a BDI programming language as a special type of belief about the action possibilities an agent can take with respect to an entity in the environment.

Algorithm 5.2 shows an agent control loop which has integrated the idea of affordances as a fundamental step in the agent reasoning model. This control loop is based on the well understood BDI based agent control loop explained by Wooldridge [179] and proposed by Rao and Georgeff [121]. This affordance based agent control loop shows how an existing agent reasoning model can be modified to incorporate the concept of affordances. After initialising the agent's set of beliefs B and intentions I with the agent's initial beliefs B_0 and initial intentions I_0, the agent reasoning model takes the following steps.

1. Receive percepts from the world.
2. Revise its internal beliefs based on these percepts.
3. Based on the beliefs about what it can see and its current intentions, finds what affordances are available.
4. Based on the available affordances and current intentions, determines the next course of action resulting in the agent's intentions being updated.
5. Executes its current intentions.

An affordance based agent reasoning model means that affordances are treated as just another type of mental attitude, such as a belief, desire or intention. In this type of

Algorithm 5.2 Affordance Based Agent Control Loop

```
1: procedure AGENT
2:     $B = B_0$
3:     $I = I_0$
4:     while true do
5:         $\rho \leftarrow perceive\_world();$
6:         $B \leftarrow revise\_beliefs(B, \rho);$
7:         $A \leftarrow find\_affordances(B, I);$
8:         $I \leftarrow course\_of\_action(I, A);$
9:         $execute\_intention(I);$
10:    end while
11: end procedure
```

framework affordances may also be thought of as a special type of belief; that is a belief about the action possibilities the agent can take with respect to some entity it has perceived in the world. The advantage of this type of approach is that it is relatively straight forward to incorporate the concept of affordance as a mental attitude into existing agent reasoning models and implemented with some agent programming languages.

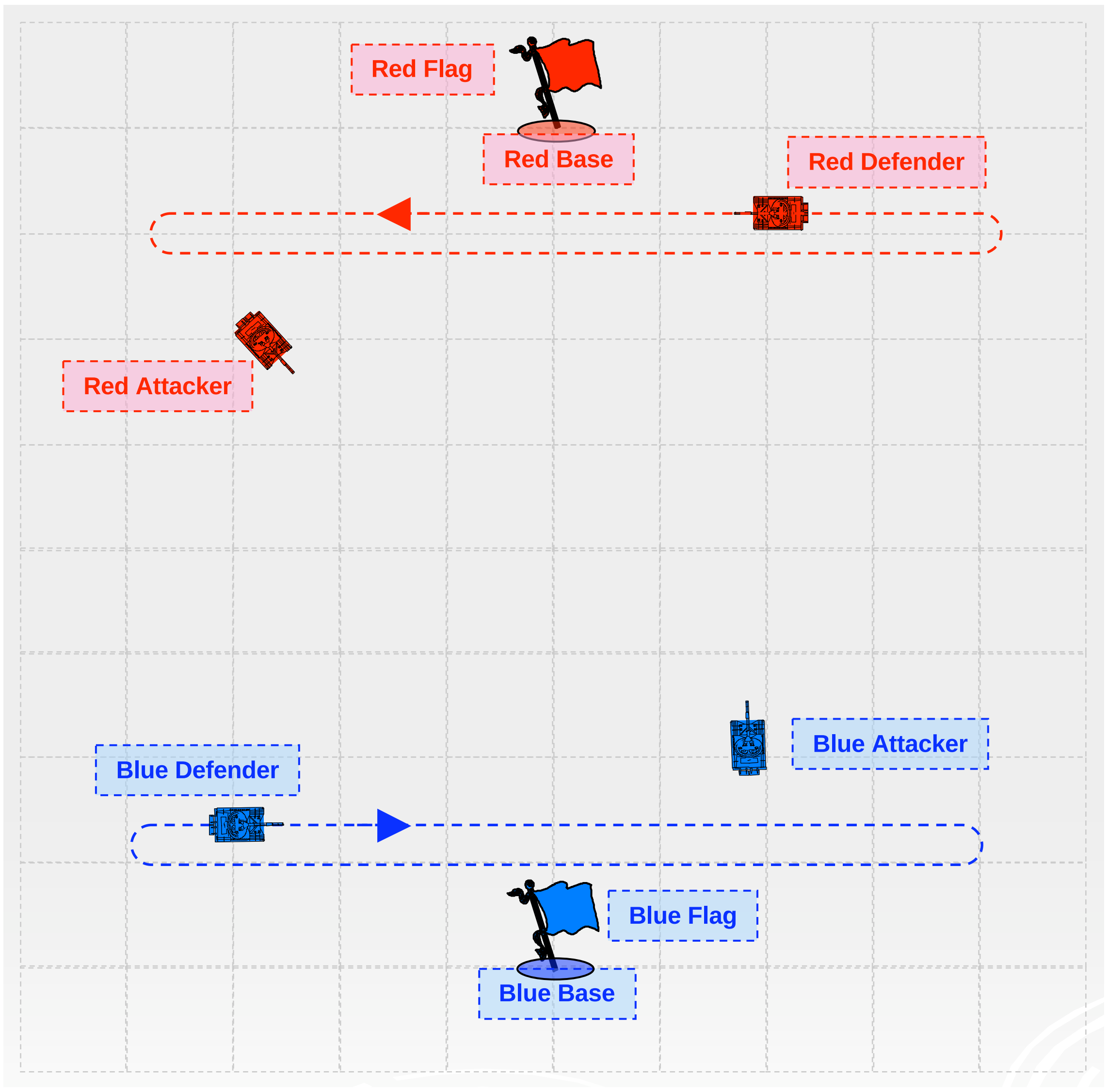
However, modelling affordances as part of an agent's internal mental state deviates from affordance theory as described in ecological psychology for a number of reasons. First, affordances are generally accepted not to be constructs of the agent or the environment but rather they exist as a result of the interaction between the two. Second, each agent can directly perceive affordances in the environment. Affordances as agent mental constructs completely bypass the concept of direct perception.

Although such an implementation of affordance does not adhere closely to the psychological theory, there may be cases and specific application domains where it is appropriate to model an agent's action possibilities in such a manner. These may be cases where cognitive plausibility is not a driving factor and other practical software engineering issues (such as integration with existing agent models) are important.

5.8 Illustrative Examples from Capture The Flag

Examples from the game of Capture The Flag (CTF) [6] will be used to illustrate concepts, ideas and examples relating to modelling agent – environment interaction using affordances. In its most simplest form, the game of Capture The Flag typically consists of two teams of players each with a flag located at its home base.

The objective for each team in the game is to capture an enemy flag and return it to its home base, while simultaneously defending its own flag from being captured by the opposing team. In most variants of the game opposing players can be removed from play by either being shot or being tagged in some way, in order to prevent them from capturing your team's flag or to allow you to capture the opposition's flag. A successful flag capture earns a team a point. The team that reaches a pre-determined number of points or who has the highest number of points after a pre-determined time period is considered the



Figures 5.3: An example scenario from a game of Capture The Flag (CTF). Each side has two tanks. Each tank undertakes a defensive and an attacking role. Each tank driver agent will perceive different affordances with respect to the other entities in the environment (such as other tanks, flags and bases). The affordances perceived will not only depend on each tanks relations with other entities in the environment but will also be influenced by the tanks role and the goals it has during the game.

winner of the game.

While a number of variants of Capture The Flag exist (both in the real and virtual worlds), the variant that is used in this thesis is a simple tank based one. This variant is played on a large square map with two teams (blue and red). The map consists of a blue and red base which hold the initial location of the blue and red team flags respectively.

Each team starts with a number of tanks, with each tank having the ability to drive around the map, perceive the environment, shoot and destroy other tanks, as well as pick up flags (either capturing an enemy flag or rescuing a friendly flag from enemy capture). The players in the game are represented by tank drivers which each operate a single tank. Each tank driver may operate a tank in roles such as attack or defence within their team.

In this thesis, this tank based game of Capture The Flag is represented and modelled as a multi-agent simulation. Each tank driver is represented as an agent, embodied by a representation of a tank and situated in a virtual environment representing the playing field consisting of other tanks, flags and bases.

The relationship between these agents and the environment which they are situated will be used to illustrate the underlying principles of affordance based agent – environment interaction throughout the thesis. This is most evident in Chapter 6, which describes the development and analysis of a multi-agent simulation of a game of Capture The Flag.

The game of Capture The Flag was selected as a mechanism for discussing affordance based agent – environment interaction for a number of reasons.

1. While a game, CTF has many similarities to the multi-agent simulations used in military operations research, such as the its dynamic, team based, adversarial and continuous nature.
2. CTF is simple enough to explain the essence of the agent – environment interaction being modelled without the complexity of domain specific knowledge required in understanding military operations research simulations.
3. The environment in which it is set in is dynamic and complex enough to allow for interesting agent-agent and agent – environment interactions to be modelled.

The Capture The Flag examples that will be considered throughout the thesis will relate to the affordances that a single tank can perceive with respect to the various entities on the game map. For example, at any given time in the game a number of other entities may be visible to on of the tanks. These entities may include an enemy tank, a friendly tank and an enemy flag that has been dropped on the ground. Depending on the situation, the different visible entities afford the tank driver different courses of actions – in other words different opportunities for action or affordances. The enemy tank may afford shooting, the enemy flag may afford being captured while the friendly tank might afford being avoided.

With a number of possible affordances available, the tank driver agent must decide which course of action or affordance to adopt. This thesis is primarily concerned with modelling the agent – environment interaction that gives rise to these affordances and the subsequent decision which must be made by an agent as to which affordance to adopt.

5.8.1 CTF Specific Definitions

In addition to the earlier specification of the affordance model in Z, it is useful to make some additional definitions that are specific to the Capture The Flag domain [54] that can be used in the context of illustrative examples.

In this variant of CTF there are only three types of entities – tanks, flags and bases. Tanks are the only type of agent entity and therefore are the only ones that can perceive affordances with respect to other entities in the environment. The types of entities can be captured in a free type definition as follows.

$$\mathit{EntityType} ::= \mathit{Tank} \mid \mathit{Flag} \mid \mathit{Base}$$

This allows for the definition of a function which determine if an entity is of a particular type.

$$\mathit{IsType} : (\mathit{Entity} \leftrightarrow \mathit{EntityType}) \rightarrow \mathbb{B}$$

The function *IsType* returns true [55] if the *Entity* is of type *EntityType*. For example, if one considers a tank called *Cobra*, then the following statement can be made.

$$\mathit{IsType}(\mathit{Cobra}, \mathit{Tank}) = \text{true}$$

It is important to know if a particular entity is of a particular type because this determines the affordances that an agent can perceive with respect to that entity. A tank agent will perceive different affordances with respect to another tank or a flag.

Entities such as tanks cannot only be killed during a game, but can also re-spawn after a period of time. Therefore it is important to know if an entity is alive or dead in order to determine if an agent should consider affordances with respect to it. The status of an entity can be defined as a free type as follows.

$$\mathit{EntityStatus} ::= \mathit{Alive} \mid \mathit{Dead}$$

This allows for the definition of a function to determine if a specific entity is alive or dead.

$$\mathit{HasStatus} : (\mathit{Entity} \leftrightarrow \mathit{EntityStatus}) \rightarrow \mathbb{B}$$

For example if there are two tanks on blue side, Cobra being alive and Mamba dead then the following statements can be made.

$$\mathit{HasStatus}(\mathit{Cobra}, \mathit{Alive}) = \text{true}$$
$$\mathit{HasStatus}(\mathit{Mamba}, \mathit{Dead}) = \text{true}$$

The affordances that a tank can perceive with respect to other tanks in the game will be directly influenced by the side of the perceived tank. That is, the affordances will

[54] Note that these definitions are in the context of the Capture The Flag domain as defined and implemented in this thesis. The definitions cannot be applied to all variations of Capture The Flag. Rather they are applicable to the specific variation of the game using tanks that is used in this thesis to illustrate simple examples of how affordances can be used in a multi-agent simulation.

[55] The Z notation for the set of boolean values is denoted by $\mathbb{B}$.

differ depending on whether the perceived tank is a friendly or enemy one. This is a good example of how social and organisational information is important in the determination of affordances. Therefore it is important to know if a particular entity (whether it be tank, flag or base) is on the same side or on the opposing side when considering the affordances for an agent. Two functions are defined to query whether two entities are on the same or opposing sides.

$$\begin{aligned}&\mathit{IsOnSameSide} : (\mathit{Entity} \leftrightarrow \mathit{Entity}) \rightarrow \mathbb{B}\\&\mathit{OnOpposingSides} : (\mathit{Entity} \leftrightarrow \mathit{Entity}) \rightarrow \mathbb{B}\end{aligned}$$

For example, if *Cobra* and *Mamba* are tanks both on blue side, then the following statements can be made.

$$\begin{aligned}&\mathit{IsOnSameSide}(\mathit{Cobra}, \mathit{Mamba}) = \text{true}\\&\mathit{OnOpposingSides}(\mathit{Cobra}, \mathit{Mamba}) = \text{false}\end{aligned}$$

Physical relations between entities are also important in ascertaining affordances. While there are many possible examples of important physical relations in the game of Capture The Flag, there are only a few that are relevant to the examples in this Chapter. These include knowing if one entity is visible to another entity, knowing if one entity is being tracked by another tank's sensors, and knowing if a particular tank happens to be the closest enemy tank. These relationships are captured in the following functions.

$$\mathit{IsVisible} : (\mathit{Entity} \leftrightarrow \mathit{Entity}) \rightarrow \mathbb{B}$$

$$\begin{array}{|l}\mathit{HasCurrentTrack} : (\mathit{Entity} \leftrightarrow \mathit{Entity}) \rightarrow \mathbb{B}\\\hline \forall\, \alpha, \varepsilon : \mathit{Entity};\ \beta : \mathbb{B} \mid\\\quad \mathit{IsType}(\alpha, \mathit{Tank}) = \text{true} \bullet \mathit{HasCurrentTrack}(\alpha, \varepsilon) = \beta\end{array}$$

$$\begin{array}{|l}\mathit{IsClosestEnemyTank} : (\mathit{Entity} \leftrightarrow \mathit{Entity}) \rightarrow \mathbb{B}\\\hline \forall\, \alpha, \varepsilon : \mathit{Entity};\ \beta : \mathbb{B} \mid\\\quad \mathit{IsType}(\alpha, \mathit{Tank}) = \text{true} \wedge \mathit{IsType}(\varepsilon, \mathit{Tank}) = \text{true} \bullet\\\qquad \mathit{IsClosestEnemyTank}(\alpha, \varepsilon) = \beta\end{array}$$

The functions *HasCurrentTrack* and *IsClosestEnemyTank* are specified as axiomatic definitions with constraints placed on the type of the entities being considered.

In a dynamic game such as Capture The Flag, other types of relations are also important in determining affordances. For example, one might need to know if an enemy tank is being targeted, is being intercepted or has a firing solution [56] available in order to determine if the enemy tank affords being destroyed. Knowledge about these relationships can be captured in the following functions.

$$\begin{array}{|l}\mathit{IsTargeting} : (\mathit{Entity} \leftrightarrow \mathit{Entity}) \rightarrow \mathbb{B}\\\hline \forall\, \alpha, \varepsilon : \mathit{Entity};\ \beta : \mathbb{B} \mid\\\quad \mathit{IsType}(\alpha, \mathit{Tank}) = \text{true} \wedge \mathit{IsType}(\varepsilon, \mathit{Tank}) = \text{true} \bullet\\\qquad \mathit{IsTargeting}(\alpha, \varepsilon) = \beta\end{array}$$

[56] A firing solution typically denotes that some enemy entity is within range of a weapon and an effective shot can be taken.

$$\begin{array}{|l}
\mathit{IsIntercepting} : (\mathit{Entity} \leftrightarrow \mathit{Entity}) \rightarrow \mathbb{B} \\
\hline
\forall\, \alpha, \varepsilon : \mathit{Entity}\beta : \mathbb{B} \mid \\
\qquad \mathit{IsType}(\alpha,\, \mathit{Tank}) = \mathrm{true} \wedge \mathit{IsType}(\varepsilon,\, \mathit{Tank}) = \mathrm{true} \bullet \\
\qquad\qquad \mathit{IsIntercepting}(\alpha, \varepsilon) = \beta
\end{array}$$

$$\begin{array}{|l}
\mathit{HasFiringSolution} : (\mathit{Entity} \leftrightarrow \mathit{Entity}) \rightarrow \mathbb{B} \\
\hline
\forall\, \alpha, \varepsilon : \mathit{Entity};\ \beta : \mathbb{B} \mid \\
\qquad \mathit{IsType}(\alpha,\, \mathit{Tank}) = \mathrm{true} \wedge \mathit{IsType}(\varepsilon,\, \mathit{Tank}) = \mathrm{true} \bullet \\
\qquad\qquad \mathit{HasFiringSolution}(\alpha, \varepsilon) = \beta
\end{array}$$

Given the nature of this game, it is expected that the status of the two flags in the game have some influence in the affordances that each of the tank agents can perceive. Each flag can be in one of a number of states. Initially each flag starts at the home base of the team it belongs to. It can be carried by a tank and if that tank is destroyed then the flag is dropped on the map to be picked up by another tank. These three states that each of the flags can be in can be captured in a free type definition.

$$\mathit{FlagState} ::= \mathit{AtHomeBase} \mid \mathit{CarriedByTank} \mid \mathit{DroppedOnMap}$$

This allows for the function *FlagMode* to be defined which returns true if the flag is in the specified state. Note that in this axiomatic definition, the entity type is constrained to be of type *Flag* because it only makes sense to query the flag state of a flag entity.

$$\begin{array}{|l}
\mathit{FlagMode} : (\mathit{Entity} \leftrightarrow \mathit{FlagState}) \rightarrow \mathbb{B} \\
\hline
\forall\, \varepsilon : \mathit{Entity};\ f : \mathit{FlagState};\ \beta : \mathbb{B} \mid \\
\qquad \mathit{IsType}(\varepsilon, \mathit{Flag}) = \mathrm{true} \bullet \mathit{FlagMode}(\varepsilon, f) = \beta
\end{array}$$

For example, if the red flag is located at the red home base, then the following statement can be made.

$$\mathit{FlagMode}(\mathit{RedFlag}, \mathit{AtHomeBase}) = \mathrm{true}$$

Two of the most important considerations when attempting to determine affordances are the intentions and the capability of the agent. As described previously taking an agent's intention into account is important due to the intentional nature of affordances. An affordance cannot be perceived if an agent does not have the capability to perform the possible action. In order to capture both the physical embodied and mental capability of an agent the set of capabilities is introduced.

$$[\mathit{Capability}]$$

This now allows for the specification of two functions, one to determine if a particular agent has a particular capability or intention. In both these axiomatic definitions the constraint is that the entity is a member of the set *Agent* because only agents can have capabilities and intentions. [57]

[57] The constraint is defined as the entity needing to be a member of the set *Agent*. However, it could also be defined as the entity needing to be of type *Tank* such that $\mathit{IsType}(\varepsilon,\ \mathit{Tank}) = \mathrm{true}$ holds, as was done in the previous axiomatic definitions in this Section. However, the more general case of membership of the set *Agent* was chosen because it allows for the possibility of other non-tank agents which have intentions and capabilities to be introduced into the game of CTF.

$$
\begin{array}{|l}
\mathit{HasCapability} : (\mathit{Entity} \leftrightarrow \mathit{Capability}) \rightarrow \mathbb{B} \\
\hline
\forall\, \varepsilon : \mathit{Entity};\ c : \mathit{Capability};\ \beta : \mathbb{B} \mid \\
\qquad \varepsilon \in \mathit{Agent} \bullet \mathit{HasCapability}(\varepsilon, c) = \beta
\end{array}
$$

$$
\begin{array}{|l}
\mathit{HasIntention} : (\mathit{Entity} \leftrightarrow \mathit{Intention}) \rightarrow \mathbb{B} \\
\hline
\forall\, \varepsilon : \mathit{Entity};\ i : \mathit{Intention};\ \beta : \mathbb{B} \mid \\
\qquad \varepsilon \in \mathit{Agent} \bullet \mathit{HasIntention}(\varepsilon, i) = \beta
\end{array}
$$

In order for an entity to afford a particular course of action for a tank in a CTF game, some set of conditions must hold. The above definitions can be used to specify the set of conditions that must hold for the affordance to be available as an agent. The function affords can be defined to determine if a particular entity provides a particular affordance to an agent.

$$
\begin{array}{|l}
\mathsf{affords} : (\mathit{Agent} \leftrightarrow \mathit{Entity} \leftrightarrow \mathit{Affordance}) \rightarrow \mathbb{B}
\end{array}
$$

A small subset of all the affordances that can exist in a game of CTF will be considered as examples. For example a tank may perceive that a dropped flag may afford rescuing or seizing. In another case an enemy tank may afford being intercepted, having a firing solution obtained against it or being destroyed. These examples of affordances in CTF will be explored in more detail in the following Section. They can considered to be members of the set *Affordance*.

$$
\begin{array}{l}
\{\mathit{RescueFlag}, \mathit{SeizeFlag}, \mathit{Intercept}, \\
\qquad \mathit{ObtainFiringSolution}, \mathit{DestroyTank}\} \in \mathit{Affordance}
\end{array}
$$

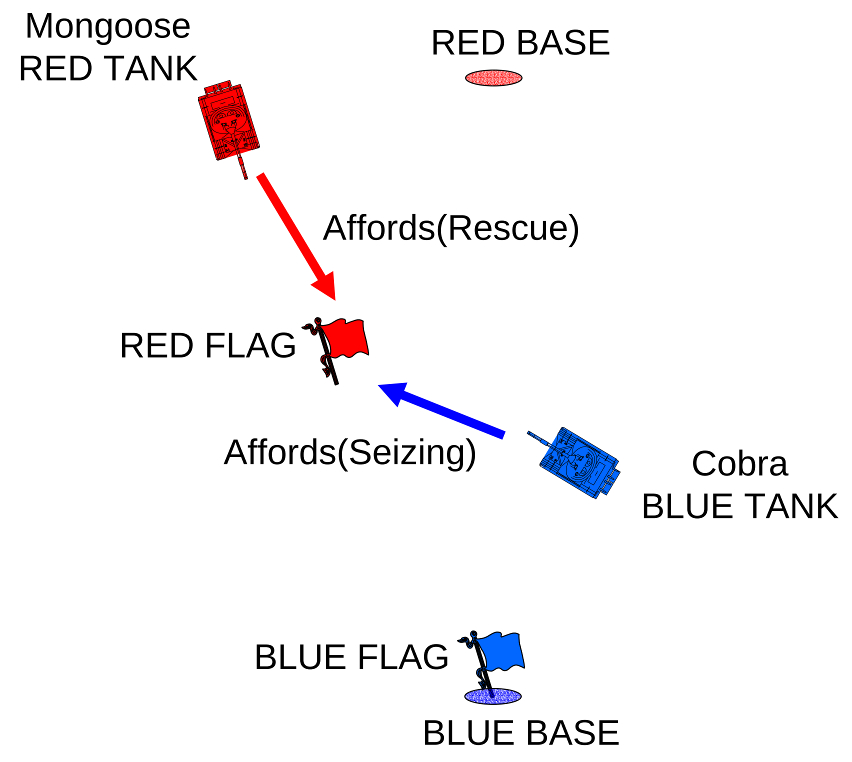
These affordances are associated with capabilities that tank agents can have. That is, an agent can not perceive an affordance if it does not have the corresponding embodied capability to adopt the possible action. The corresponding action possibilities being considered in the following examples are members of the set *Capability*.

$$
\begin{array}{l}
\{\mathit{RescueFlagCapability}, \\
\qquad \mathit{SeizeFlagCapability}, \\
\qquad \mathit{InterceptCapability}, \\
\qquad \mathit{ObtainFiringSolutionCapability}, \\
\qquad \mathit{DestroyTankCapability}\} \in \mathit{Capability}
\end{array}
$$

5.8.2 Example: Rescuing or Seizing a Flag

Consider the case in a CTF game where a red flag has been dropped on the playing field as illustrated in Figure 5.4. Two tanks, one blue and one red can both perceive the flag. Both tanks can perceive that the entity in front of them is a flag and that its colour is red. Although both tanks can perceive the same entity, it has different meaning for each of them. Hence, although both tank driving agents can perceive the same flag, with the same attributes, the affordances each agent perceives with respect to the flag are different.

Rescuing Flag and Seizing Flag Affordances at time $t = t_0$.



$$\begin{array}{l}\forall\,\alpha : \mathit{Agent};\ \varepsilon : \mathit{Entity}\ | \\ \quad \mathit{IsType}(\alpha, \mathit{Tank}) \wedge \mathit{HasStatus}(\alpha, \mathit{Alive}) \wedge \\ \quad \mathit{IsType}(\varepsilon, \mathit{Flag}) \wedge \mathit{IsVisible}(\alpha, \varepsilon) \wedge \\ \quad \mathit{IsOnSameSide}(\alpha, \varepsilon) \wedge \mathit{HasCurrentTrack}(\alpha, \varepsilon) \wedge \\ \quad \mathit{FlagMode}(\varepsilon, \mathit{DroppedOnMap}) \wedge \\ \quad \mathit{HasCapability}(\alpha, \mathit{RescueFlagCapability}) \\ \bullet\ \mathsf{affords}(\alpha, \varepsilon, \underline{\mathit{RescueFlag}})\end{array}$$

$$\begin{array}{l}\forall\,\alpha : \mathit{Agent};\ \varepsilon : \mathit{Entity}\ | \\ \quad \mathit{IsType}(\alpha, \mathit{Tank}) \wedge \mathit{HasStatus}(\alpha, \mathit{Alive}) \wedge \\ \quad \mathit{IsType}(\varepsilon, \mathit{Flag}) \wedge \mathit{OnOpposingSides}(\alpha, \varepsilon) \wedge \\ \quad \mathit{FlagMode}(\varepsilon, \mathit{DroppedOnMap}) \wedge \mathit{IsVisible}(\alpha, \varepsilon) \wedge \\ \quad \mathit{HasCurrentTrack}(\alpha, \varepsilon) \wedge \\ \quad \neg\,\mathit{HasIntention}(\alpha, \mathit{RescueFlag}) \wedge \\ \quad \mathit{HasCapability}(\alpha, \mathit{SeizeFlagCapability}) \\ \bullet\ \mathsf{affords}(\alpha, \varepsilon, \underline{\mathit{SeizeFlag}})\end{array}$$

Figures 5.4: Rescuing and Seizing Flag Affordances. The CTF multi-agent simulation at time t_0.

In this example as illustrated in Figure 5.4, for the red tank the flag affords rescuing to prevent any enemy tanks from obtaining it. On the other hand, for the blue tank the flag affords seizing so that it can be taken to the blue base and captured and subsequently a point scored.

Figure 5.4 also declares as Z statements the conditions that must hold for an entity ε to afford a course of action to an agent α.

In the case of *Cobra*, (the blue tank) it can perceive a an entity, which only affords being seized if it meets a number of conditions which include the entity being a flag, being on the opposing team, it has been dropped on the map and it is visible and being tracked. Note that the agent must also have the capability to be able to seize the flag. Importantly an enemy flag only affords being seized if the tank does not have the intention to rescue a flag. This is indicated by the predicate $\neg\, \mathit{HasIntention}(\alpha, \mathit{RescueFlag})$. This is because if a tank is in the process of rescuing its own flag, then that takes priority over seizing the enemy flag.

In this particular example, *Cobra* can only perceive one affordance with respect to the red flag. This can be specified with the affords predicate.

$$\mathsf{affords}(\mathit{Cobra}, \mathit{RedFlag}, \mathit{SeizeFlag})$$

Alternatively using the notation from Section 5.4 the set of affordances that *Cobra* can perceive with respect to *RedFlag* at time t_0 can be specified as follows.

$$\Phi_{\mathit{Cobra},\mathit{RedFlag}}(t_0) = \{\mathit{SeizeFlag}\}$$

Figure 5.4 also illustrates an affordance relationship between the red tank *Mongoose* and the red flag. The figure declares the set of conditions that must hold for an entity to perceive the affordance *RescueFlag* with respect to another entity. For example, a flag can only be rescued by a tank if it is on the same side and it has been dropped on the map. As for the case of the blue tank *Cobra*, the affordance relation between *Mongoose* and the red flag can be specified as follows.

$$\mathsf{affords}(\mathit{Mongoose}, \mathit{RedFlag}, \mathit{RescueFlag})$$

Alternatively using the set notation from Section 5.4 the affordance relation can be specified as follows.

$$\Phi_{\mathit{Mongoose},\mathit{RedFlag}}(t_0) = \{\mathit{RescueFlag}\}$$

This allows for the entire set of affordances in the CTF multi-agent simulation at time t_0 to be specified as follows.

$$\begin{aligned}\boldsymbol{\Phi}_{\mathbf{CTF}}(t_0) \;=\; & \{\mathsf{affords}(\mathit{Cobra}, \mathit{RedFlag}, \mathit{SeizeFlag}),\\ & \mathsf{affords}(\mathit{Mongoose}, \mathit{RedFlag}, \mathit{RescueFlag})\}\end{aligned}$$

5.8.3 Example: Intercepting a Tank

The second example relates to an attempt by a tank driver to intercept and destroy an enemy tank. While the previous example illustrated how observers can perceive different affordances with respect to the same entity, this example shows how the affordances with respect to the same entity change over time.

The interception of an enemy tank has been implemented as a three step process. The steps correspond to the different affordances that the intercepting tank perceives with respect to the enemy tank as the situation changes. The steps are as follows.

- Determine if the enemy tank afford being intercepted?
- If it does afford being intercepted then does it afford having a firing solution be obtained against it. A firing solution involves the intercepting tank getting close enough to the enemy tank to be within firing range of a weapon.
- Once a firing solution has been obtained, then the enemy tank affords being destroyed and the weapon can be fired.

5.8.3.1 Intercept Affordance

A tank only affords being intercepted by another tank under a very strict set of conditions. An illustrative example and the set of conditions which must be satisfied for an *Intercept* affordance to be available are shown as a Z statement in Figure 5.5.

In this example, at a time t_1 in the game, the blue tank *Cobra* has detected the red tank *Panther*. In order for the *Intercept* affordance to become available for *Cobra* with respect to *Panther*, the two tanks need to be on opposing sides. Additionally *Panther* must be the closest enemy tank to *Cobra*. This is the case in Figure 5.5.

At time t_1 the affordances in the multi-agent simulation can be specified as follows.

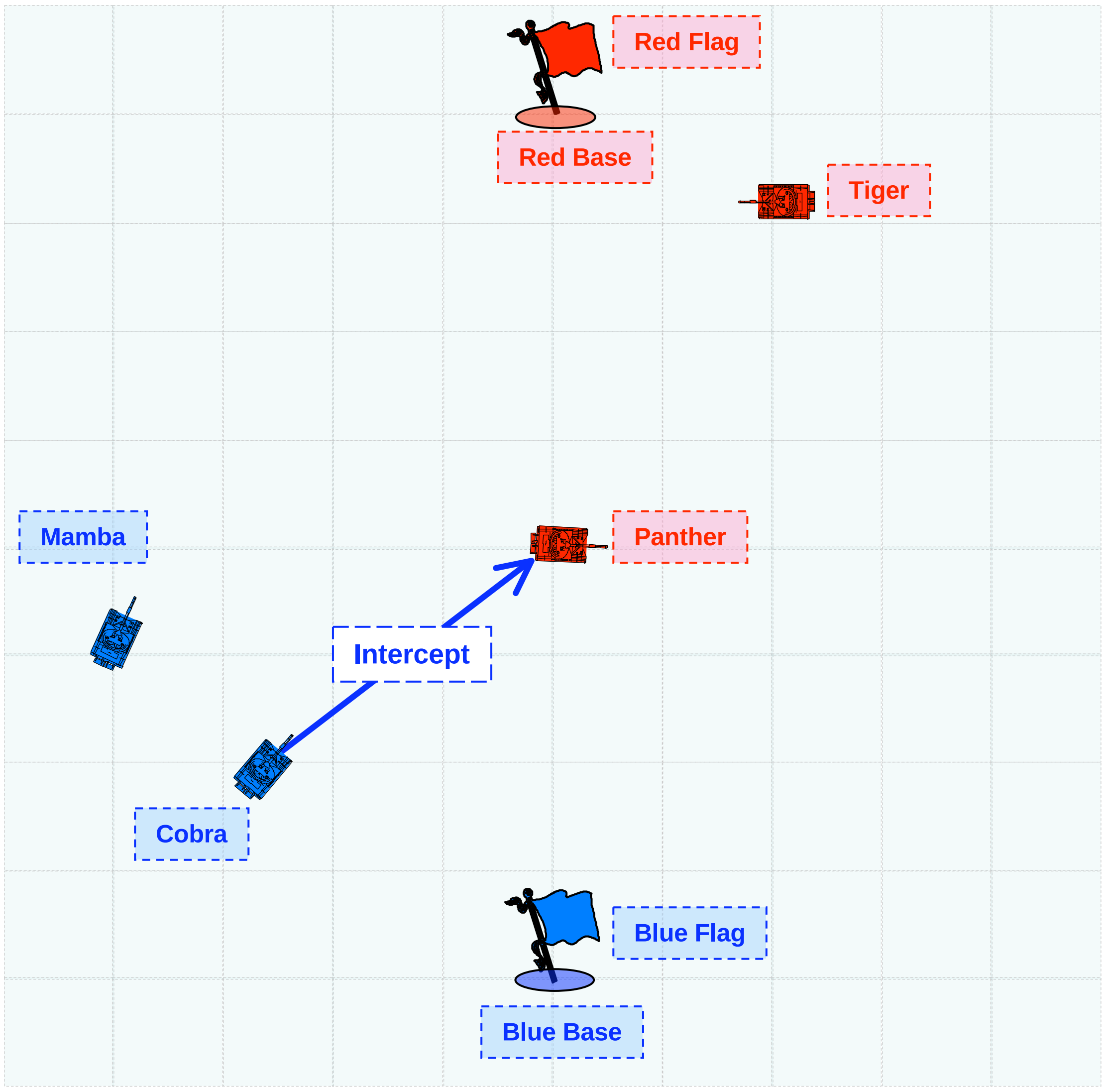
$$\mathbf{\Phi_{CTF}}(t_1) = \{\mathsf{affords}(\mathit{Cobra}, \mathit{Panther}, \mathit{Intercept})\}$$

Perhaps the two most important conditions for the interception affordance to hold is that the intercepting tank must have the capability to intercept and its current intention is to defend its home base. The reason that these two conditions are important is that they are central to the concept of affordance used in this game.

The condition associated with the agent's current intention is important because it relates to the intentional nature of affordance. In this particular case an enemy tank only affords being intercepted if the current tank's intention is to defend its home base.

This can be compared to the case of a tank attacking the enemy base. Even if it encounters an enemy tank and all the other conditions are met, that enemy tank does not necessarily afford being intercepted if the tank driver's intention is to attack and capture the enemy flag as opposed to defending the home base. This differentiation underscores the importance of intention in determining affordances.

Intercept Affordance at time $t = t_1$.



$$\begin{array}{l}
\forall\, \alpha : \mathit{Agent};\ \varepsilon : \mathit{Entity} \mid \\
\quad \mathit{IsType}(\alpha, \mathit{Tank}) \wedge \mathit{HasStatus}(\alpha, \mathit{Alive}) \wedge \\
\quad \mathit{IsType}(\varepsilon, \mathit{Tank}) \wedge \mathit{HasStatus}(\varepsilon, \mathit{Alive}) \wedge \\
\quad \mathit{OnOpposingSides}(\alpha, \varepsilon) \wedge \\
\quad \mathit{HasCapability}(\alpha, \mathit{InterceptCapability}) \wedge \\
\quad \mathit{IsVisible}(\alpha, \varepsilon) \wedge \mathit{IsClosestEnemyTank}(\alpha, \varepsilon) \wedge \\
\quad \mathit{HasIntention}(\alpha, \mathit{DefendBase}) \\
\bullet\ \mathsf{affords}(\alpha, \varepsilon, \underline{\mathit{Intercept}})
\end{array}$$

Figures 5.5: Intercept Affordance. The CTF multi-agent simulation at time t_1.

5.8.3.2 Obtain Firing Solution Affordance

Once the tank driver adopts the affordance to intercept the enemy tank, the next question is to determine if this tank affords having a firing solution obtained against it.

Figure 5.6 shows the state of the game at time t_2 where the blue tank *Cobra* can perceive an affordance to obtain a firing solution against the red tank *Panther*. The figure also declares the set of conditions which must hold for this affordance to become available for a particular agent. Note that the conditions include that the agent's current intention (in this case *Cobra*) must be to intercept and that the entity being intercepted is the one that is being considered (in this case *Panther*). These conditions are indicated by the predicates $\mathit{HasIntention}(\alpha, \mathit{Intercept}$ and $\mathit{IsIntercepting}(\alpha, \varepsilon)$. Furthermore, the predicate $\neg\, \mathit{HasFiringSolution}(\alpha, \varepsilon)$ indicates that *Cobra* must not have a firing solution against *Panther*.

Therefore at time t_2 the affordances in the multi-agent simulation can be specified as follows.

$$\mathbf{\Phi_{CTF}}(t_2) = \{\mathsf{affords}(\mathit{Cobra}, \mathit{Panther}, \mathit{ObtainFiringSolution})\}$$

Perhaps the most interesting conditions are that the tank driver's current intention conducting an intercept against a particular tank and that a firing solution has yet to be obtained.

5.8.3.3 Destroy Enemy Tank Affordance

Once a firing solution has been obtained, that is the enemy tank is within weapon's range then another affordance becomes available. At time t_3 in the game *Panther* affords being destroyed by *Cobra*. This situation and the set of conditions which must hold for this affordance to be perceived by an agent in the game are illustrated in Figure 5.7.

Therefore at time t_3 the affordances in the multi-agent simulation can be specified as follows.

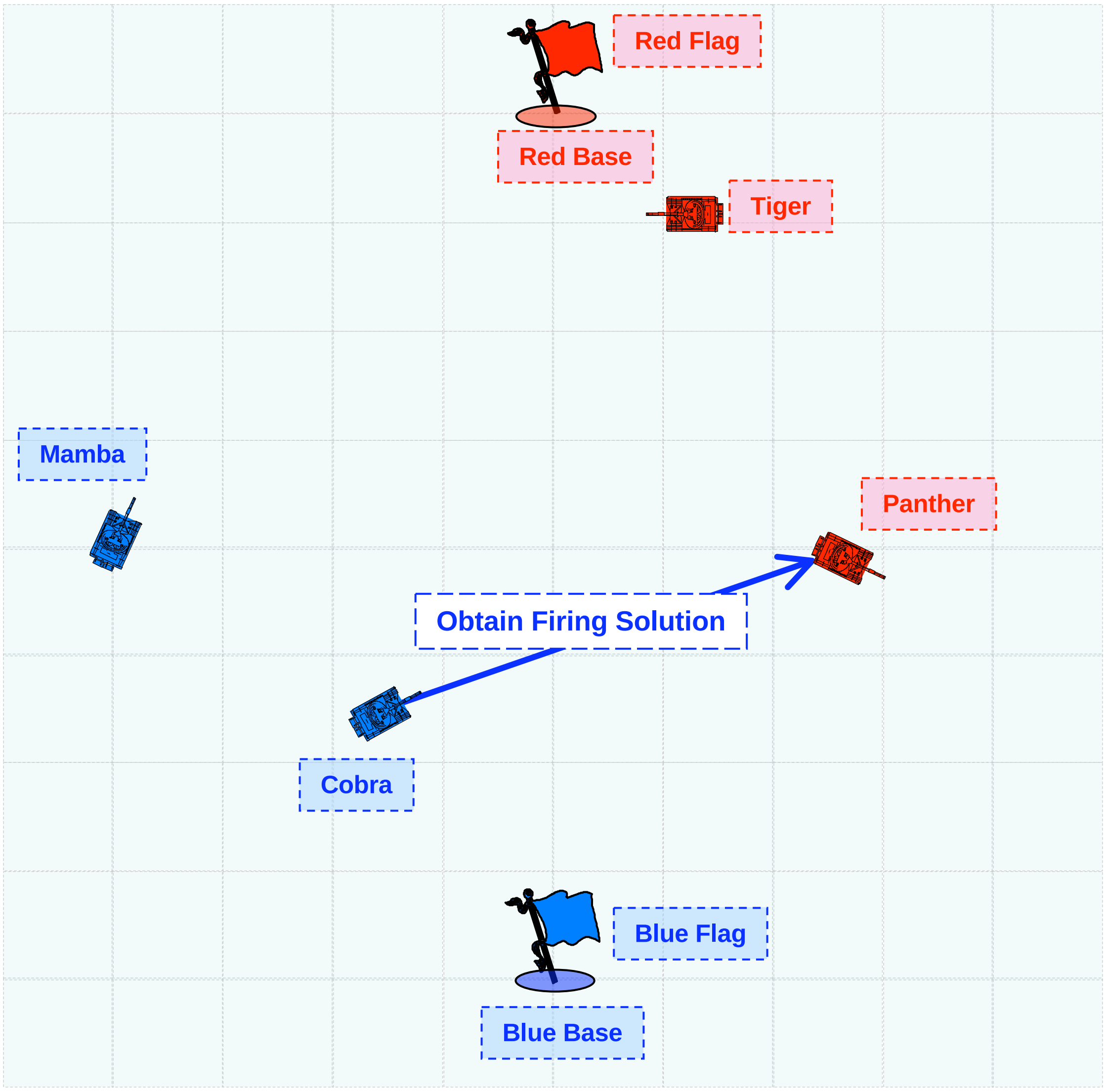
$$\mathbf{\Phi_{CTF}}(t_3) = \{\mathsf{affords}(\mathit{Cobra}, \mathit{Panther}, \mathit{DestroyEnemyTank})\}$$

Finally, once the red tank *Panther* affords being destroyed by *Cobra*, Cobra can fire its weapon to attempt a destruction. The case where the red tank has been successfully destroyed and has been removed from the game is illustrated in Figure 5.8.

5.9 Computational Performance Implications

Consider a multi-agent system which consists of m entities, n of which are agents that can perceive affordances such that $n \leq m$. If each agent considers affordances with respect to every other entity in the simulation except itself, then there are $n(m-1)$ agent-entity interactions which the algorithm must consider. Therefore, the number of steps required to evaluate all the possible agent-entity affordances is given by $T(n, m) = n(m-1)$. Hence

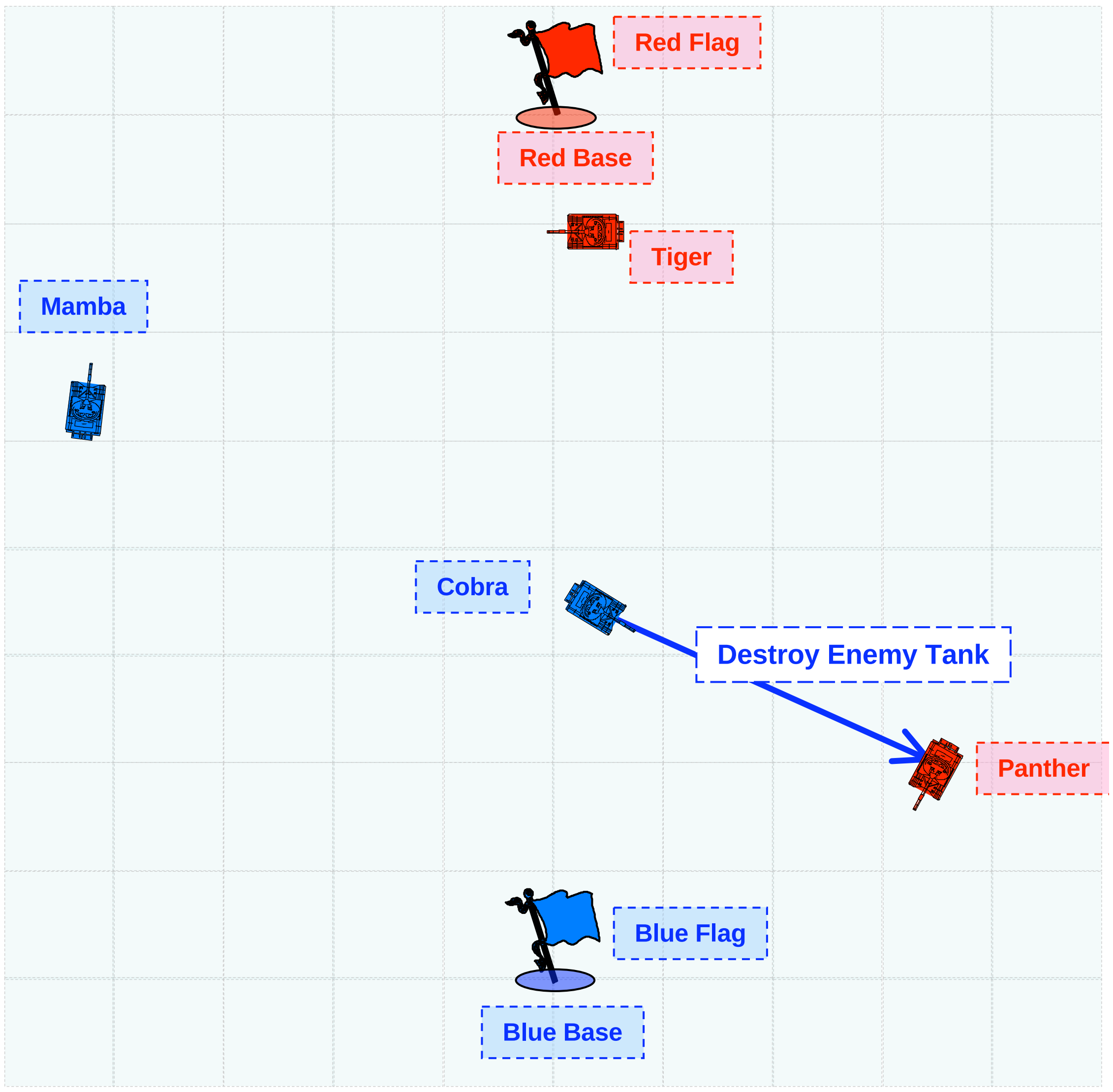
Obtain Firing Solution Affordance at time t = t_2.



$$
\begin{array}{l}
\forall\, \alpha : \mathit{Agent};\ \varepsilon : \mathit{Entity}\ | \\
\qquad \mathit{IsType}(\alpha, \mathit{Tank}) \wedge \mathit{HasStatus}(\alpha, \mathit{Alive}) \wedge \\
\qquad \mathit{IsType}(\varepsilon, \mathit{Tank}) \wedge \mathit{HasStatus}(\varepsilon, \mathit{Alive}) \wedge \\
\qquad \mathit{OnOpposingSides}(\alpha, \varepsilon) \wedge \mathit{IsVisible}(\alpha, \varepsilon) \wedge \\
\qquad \mathit{IsClosestEnemyTank}(\alpha, \varepsilon) \wedge \mathit{HasIntention}(\alpha, \mathit{Intercept}) \wedge \\
\qquad \mathit{IsIntercepting}(\alpha, \varepsilon) \wedge \neg\, \mathit{HasFiringSolution}(\alpha, \varepsilon) \wedge \\
\qquad \mathit{HasCapability}(\alpha, \mathit{ObtainFiringSolutionCapability}) \\
\bullet\ \mathsf{affords}(\alpha, \varepsilon, \underline{\mathit{ObtainFiringSolution}})
\end{array}
$$

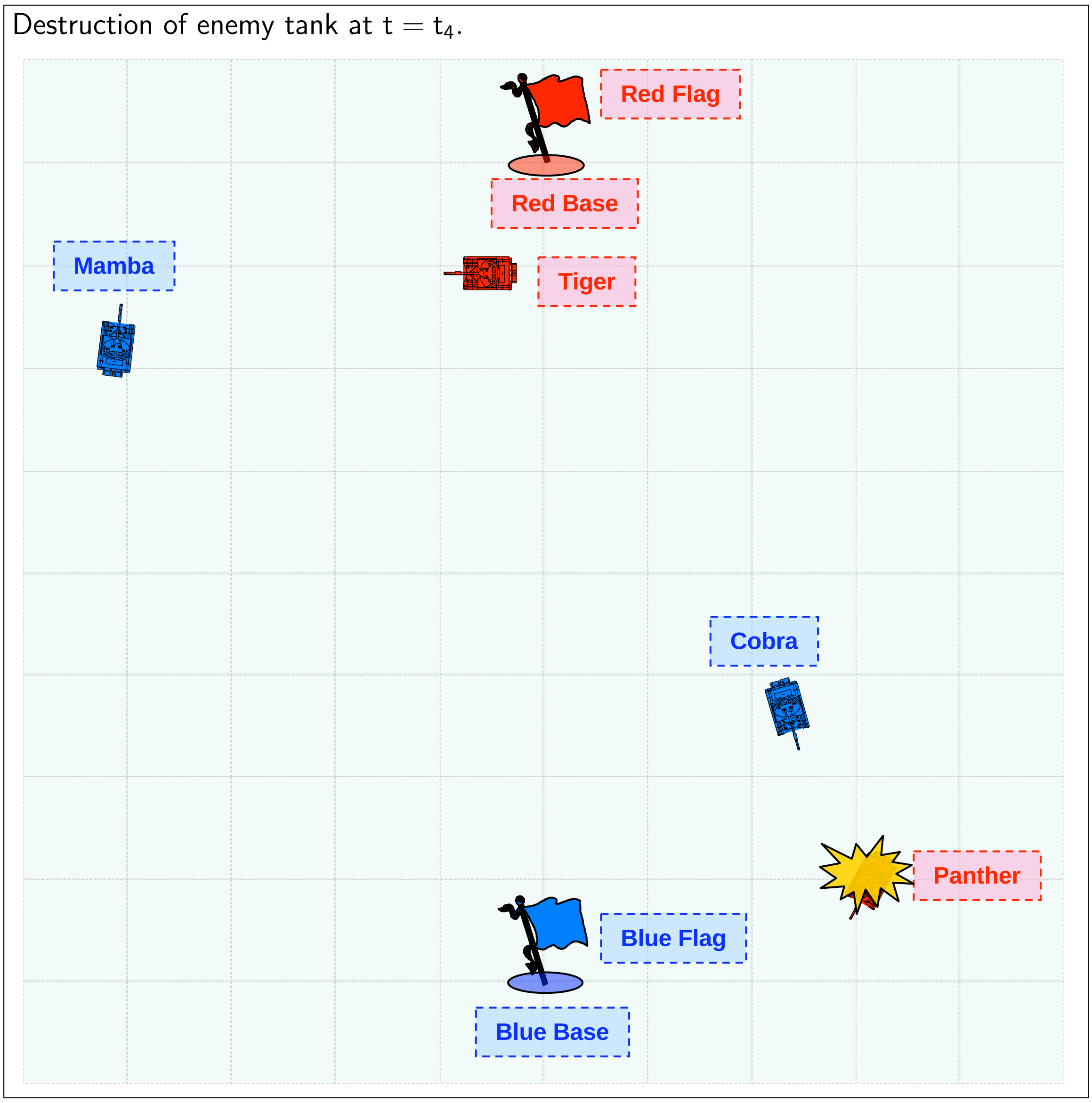
Figures 5.6: Obtain Firing Solution Affordance. The multi-agent simulation at time t_2.

Destroy Enemy Tank Affordance at t = t_3.



$$\begin{array}{l}
\forall\, \alpha : \mathit{Agent};\ \varepsilon : \mathit{Entity} \mid \\
\quad \mathit{IsType}(\alpha, \mathit{Tank}) \wedge \mathit{IsType}(\varepsilon, \mathit{Tank}) \wedge \\
\quad \mathit{HasStatus}(\alpha, \mathit{Alive}) \wedge \mathit{HasStatus}(\varepsilon, \mathit{Alive}) \wedge \\
\quad \mathit{OnOpposingSides}(\alpha, \varepsilon) \wedge \\
\quad \mathit{HasIntention}(\alpha, \mathit{ObtainFiringSolution}) \wedge \\
\quad \mathit{IsTargeting}(\alpha, \varepsilon) \wedge \mathit{HasFiringSolution}(\alpha, \varepsilon) \wedge \\
\quad \mathit{HasCapability}(\alpha, \mathit{DestroyEnemyTankCapability}) \\
\bullet\ \mathsf{affords}(\alpha, \varepsilon, \underline{\mathit{DestroyEnemyTank}})
\end{array}$$

Figures 5.7: Destroy Enemy Tank Affordance. The CTF multi-agent simulation at time t_3.



Figures 5.8: Destruction of an Enemy Tank. The CTF multi-agent simulation at time t_4.

such an algorithm has order nm complexity which can be denoted using Landau or Big O notation.

$$T(n, m) = n(m - 1) \in O(nm) \tag{24}$$

In a pure multi-agent system where all entities are agents such that $m = n$, then the complexity of the algorithm can clearly be seen to be quadratic.

$$T(n, n) = n(n - 1) = n^2 - n \in O(n^2) \tag{25}$$

For each agent-entity interaction the algorithm must then consider the set of all possible affordances p a particular agent may have with respect to that entity. Therefore the number of affordance evaluations in the algorithm is given by $T(n, m, p)$.

$$T(n, m, p) = pn(m - 1) \tag{26}$$

Although the quadratic nature of such an algorithm may seem computationally problematic, there are a number of practical optimisations which can be made to reduce the computation time.

- For the type military multi-agent simulations being considered, the number of agents and entities in the world are not very large. For example typical air combat simulations may include a total of 2, 4, 8 or 16 entities, where each entity is a fighter aircraft flown by a fighter pilot agent. [58]
- The number of agent – environment interactions being considered can be significantly reduced if only the entities in each agent's sensory field of view are evaluated. Obviously there is no reduction in the number of evaluations if each agent can see all other entities in the world. A disadvantage of this approach is that no consideration is given to affordances which may exist independent of an agent's perception of them.
- The number of affordances being considered for each agent-entity relation can be reduced by considering only the affordances which are relevant to each agent.

In the case of the Capture The Flag example being considered, the total number of entities m in the simulation is two more than the total number of agents n, accounting for the two flags and two home bases. As the number of agents n and the number of action possibilities p being considered for each agent increase, the total number of evaluations of possible actions that need to be undertaken by the simulation significantly increases.

Table 5.1 presents the number of evaluations required for 2, 4, 8, 16 and 32 agents for increasing number of action possibilities. Each evaluation dynamically considers if a particular action possibility is available for a particular agent, and hence as is shown by the table, the number of evaluations increase significantly resulting in increased computation time.

What is not indicated by Table 5.1 is the nature of these evaluations. It is important to note that not all evaluation of action possibilities are computationally equivalent. The computation time for each evaluation will depend on a number of factors.

[58]Typical air combat scenarios may involve the simulation of blue versus red fighter aircraft in 1v1, 2v2, 4v4 or 8v8 configurations. While air combat involving a larger number of fighter aircraft is certainly possible, it is rarer and less likely to be studied using multi-agent simulation approach.

Tables 5.1: Number of evaluations required as the number of agents and the number of action possibilities increase in a game of Capture The Flag.

	$n = 2$	$n = 4$	$n = 8$	$n = 16$	$n = 32$
$p = 1$	6	20	72	272	1056
$p = 5$	30	100	360	1360	5280
$p = 10$	60	200	720	2720	10560
$p = 25$	150	500	1800	6800	26400
$p = 50$	300	1000	3600	13600	52800
$p = 100$	600	2000	7200	27200	105600

Affordance Implementation Mechanism The mechanism by which affordances are implemented in a multi-agent simulation will significantly affect the computation time. If affordances are implemented as simple static annotations on entities in the world then the evaluation function is likely to be much simpler than the case where the evaluation of an action possibility is a complex function which takes a number of factors into account.

Agent Variability The amount of computation will also vary by agent type. Whereas in Capture The Flag there is only one agent type (a tank), one can imagine a multi-agent simulation with many types of agents where the determination of action possibilities varies with agent type.

The number of evaluations can be reduced in a number of ways. For example, it is possible to prioritise the entities in the environment for each agent and only consider and evaluate action possibilities for those entities that are considered high priority by each agent. Prioritisation can take many forms and may include considering only entities which are within an agent's field of view, prioritising entities based on physical distance or prioritisation by another mechanism such as the level of threat that an entity poses to an agent.

Another possibility is to divide the list of action possibilities into categories and only consider the categories which are appropriate for a particular agent type. For example, in Capture The Flag the action possibilities associated with agents perceiving tanks on their own team are different to those that they perceive with respect to tanks on the opposing team.

How one optimises the model of affordance presented in this Chapter will be significantly dependant on the particular application domain and the specific software architecture selected for a multi-agent simulation.

5.10 Summary

This Chapter has presented a model of agent – environment interaction in multi-agent simulation that has been directly inspired by theory of affordances from ecological psychology and motivated by the requirements of military multi-agent simulations.

The Chapter begins by drawing on the debate on the nature of affordances from the ecological psychology community. Using this knowledge the properties and characteristics of affordances that are most relevant to multi-agent simulation are discussed in the context of developing a computational model. The aspects of a multi-agent simulation which need to be considered for computing affordances are discussed, and a model of affordance based agent reasoning is presented.

The idea of affordances as relationships between agents and entities in the environment was presented. Subsequently an algorithm is presented outlining how these affordance relationships between agents and entities can be found in a multi-agent simulation.

A formal model of affordance based agent reasoning is then presented. The model was defined using the formal specification language Z and the different agent reasoning steps were structured using a simplified version of Boyd's OODA loop.

A number of illustrative examples from the game Capture The Flag were presented to help explain the model and ground it in a practical example of a multi-agent simulation.

Finally, the Chapter briefly discusses the computational performance implications of an affordance based approach to agent reasoning.

The model of affordance described in this Chapter was the basis of the affordance based model used in the Human Agent Virtual Environment (HAVE) described in Chapter 2. This has shown the viability of the affordance based approach in a real and complex multi-agent simulation.

However, in order to evaluate a number of different design interpretations of this model, a cut down multi-agent simulation has been developed implementing the ideas presented in this Chapter. This multi-agent simulation implements the game of Capture The Flag which was described in this Chapter and is used to empirically evaluate a number of aspects of the affordance based approach in a practical simulation.

Chapter 6

Evaluation: Capture The Flag

"An affordance is a relationship between an object in the world and the intentions, perceptions and capabilities of a person."

— Mark Weiser and John Seely Brown [170]

6.1 Introduction

In this Chapter the model of affordance based interaction described in Chapter 5 is evaluated in the context of a practical multi-agent simulation. The example application domain of Capture The Flag (CTF) which was used to illustrate aspects of the affordance model in Chapter 5 is in this Chapter, developed into a multi-agent simulation incorporating an affordance based interaction mechanism.

By incorporating the ideas from the affordance model into a realised design and implementation of a multi-agent simulation the practical issues associated with modelling affordances can be evaluated. The most important factor in the evaluation is how the affordance model is interpreted from a software architecture design perspective. Due to the high abstract nature of the model presented in Chapter 5 it is possible to interpret the model into a number of different software designs.

While the focus of the model is on the computation of affordances, it does not mandate where in a software architecture the affordances are to be computed. Hence, in this chapter the two extreme cases are considered – in the first case where the affordances are computed in the virtual environment and in the second case where affordances are computed by each agent. These interpretations correspond to the cases where the intelligent reasoning is undertaken in either the environment or in the agent. These two interpretations then take the form of different software architectural designs within the same multi-agent simulation framework.

An empirical evaluation of these interpretations is conducted by looking at how the different design approaches impact on computational performance of the multi-agent simulation. The impact of the different design decisions on the multi-agent systems is also discussed in the context of the Capture The Flag multi-agent simulation.

This Chapter begins with a short description of the aims of the evaluation. This is followed by a description of the domain of Capture The Flag and a listing of the rules of the game used in the multi-agent simulation that was developed.

A description of the CTF multi-agent simulation and the various components is set out, followed by an outline of the experiments conducted and how they were setup. The results from the experiments along with detailed discussion and analysis of the relevant issues involved are then presented.

6.2 Evaluation Aims

In Chapter 2 an affordance based model of a strike fighter pilot was described. The model implemented in HAVE was based on the affordance model described in Chapter 5. Each pilot agent in HAVE computed its own affordances. This was an example of affordance computation in the agent or what has also been referred to in this thesis as an intelligent agent. The purpose of the affordance based pilot agent in HAVE was to demonstrate the viability of an affordance based approach to agent reasoning in a real world and relatively complex application domain.

However, in order to evaluate the affordance model described in Chapter 5 a broader evaluation is required. The Capture The Flag (CTF) multi-agent simulation described in this Chapter was developed to facilitate this broader evaluation.

As described in Chapter 5 it is possible to interpret the affordance model in a number of different ways in terms of the design of a multi-agent simulation. For the purposes of evaluation two different architectural design interpretations for affordance based reasoning are considered. These are:

(1) Intelligent Environment Where affordances for each agent are determined by the environment and which each agent can directly perceive in the environment.

(2) Intelligent Agent Where affordances for each agent are determined by each agent.

The CTF multi-agent simulation described in this Chapter was developed in a manner to allow for both these architectural choices to be employed within the same simulation system. The simulation allows for the architecture type to be dynamicall selected at simulation start time. This allows all other component of the simulation to remained the same regardless of the affordance mechanism being used. This approach allows for a fair comparison of of the Intelligent Environment and Intelligent Agent architectures within the same multi-agent simulation system.

The primary aim of this evaluation is to compare these two alternative architectures for affordance based agent – environment interaction within the context of a simulation game which has properties similar found in many military multi-agent simulations. Specifically, evaluation aims to compare the two architectural approaches in three main areas.

Model Fidelity: How closely each architecture type adheres to concept of affordance as described in ecological psychology.

Design Impact: The impact each architecture type has on the design of the agent, the environment the interaction between the two and the overall multi-agent simulation.

Computational Performance Impact: The impact of each architecture type on computational performance measured through a number of run-time metrics.

The evaluation of the computational performance of the various architecture types is conducted by recording the results from many different simulation runs. The metrics which are recored include:

CTF Game Metrics By recording metrics associated with the game of Capture The Flag (such as overall score, number of kills etc.) it is possible to evaluate if the game is generating reasonable results as well as providing a baseline for comparison between different architecture types.

Total Simulation Computation Time By measuring the total computation time of each simulation run it is possible to directly compare the impact each of the architecture types has on the computational performance of the multi-agent simulation. Furthermore, since it is possible to record the amount of computation time spent in each function or method call, it is possible to conduct further analysis and evaluate how much computation time is being spent in all the major subsystems of the simulation.

Number of Function and Method Calls By measuring the number of function and method calls in each simulation run, it is possible to get an indirect measure of the complexity of each of the architecture types being considered.

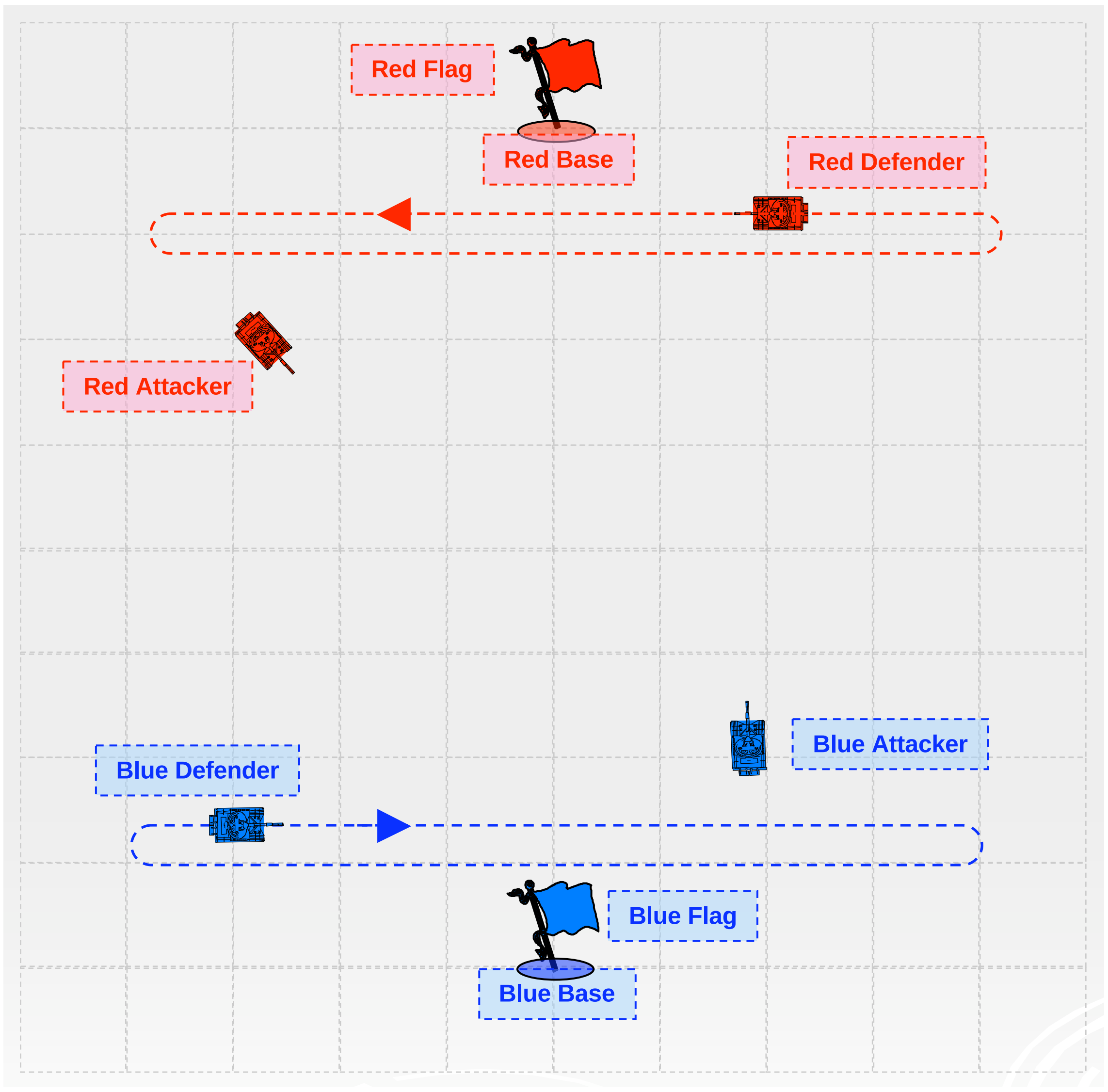
These metrics are described in further detail in Section 6.7 in the context of the experiments being undertaken.

6.3 The Capture The Flag Domain

In Section 5.8 a brief description of the game Capture The Flag was presented together with a number of illustrative examples. In this Section the specific rules of the game used for the simulation and evaluation are described in more detail.

While many variants of CTF exist, in this thesis a tank based variant is used where the tanks are the agents in the game. More precisely, the tank and its associated systems (vehicle, sensor and weapon) provide the embodiment for the tank driver, which can be considered as a cognitive or decision making model. It is the decision making abilities of components such as the tank driver in this multi-agent simulation that are often referred to as intelligent agents.[59]

[59]While the in many multi-agent simulations the entire tank would be considered the agent, it is sometimes important to make a distinction between the physical and cognitive parts of the entire system considered as a tank.



Figures 6.1: A typical layout of a CTF game map as used in the CTF multi-agent simulation. The game map consists of a square playing field with a red base located in the northern part of the map, while a blue base is located in the southern part of the map. The game starts with each side's flag located at the respective home base. In the scenario indicated each side consists of two tanks, one acting in the role of a defender (whose job is to prevent enemy tanks obtaining the team flag) and one in the role of an attacker whose job is capture the enemy's flag and bring it back to the home base to score a point. The CTF multi-agent simulation models the affordance based interaction between the tanks (as agents) and other entities in the virtual environment. The dotted lines on the game map indicate the patrol paths for the defending tanks.

While it is possible to program an agent to play a game of CTF using a myriad of techniques, here the focus is on using an affordance approach. What is of interest is the type of affordances that a tank driver can perceive with respect to the other entities in the world (such as other tanks, flags and bases).

The game of Capture The Flag (CTF) typically consists of two teams of players (for example tanks), and two flags (red and blue). The game is played in a large area with each team's base at opposite ends of the playing field. Each team has a flag which is located at the home base at the start of the game.

The objective of the game is to capture the opposing team's flag and to bring it to the team's home base while simultaneously preventing the opposing team from capturing your own flag. Players can remove members of the opposing team from the game by successfully shooting at them and scoring a hit. A team scores a point, known as a flag capture, when it has both flags at its home base. The team with the most flag captures at the end of a specified time period wins the game.

Many variants of CTF exist that are played both in the real world and in the virtual world of video games [6]. A simplified variant was selected for the work described in this paper, with each player represented using a tank. A two versus two (2v2) scenario was chosen with one tank on each side undertaking the role of either attacker or defender.

The game of CTF was chosen for the evaluation because it is played in a continuous, dynamic, and adversarial environment in which the player agents were embodied in models of tanks. This made the game conceptually similar and indicative of the type of military simulations (air to air combat of two to eight fighter aircraft) which motivates this research.

Although CTF is a complex, dynamic game and can be used to evaluate implementations of affordances based architectures, it is important to note that it is not as complex as a military simulation such as HAVE described in Chapter 6.

Figure 6.1 shows the layout of the game map that is used in this Chapter. The specific rules of CTF used in the multi-agent simulation described in this Chapter are as follows.

1. The game is played on a square continuous playing field of dimensions $100m \times 100m$. All players are confined to within the playing field.
2. There are three types of entities on the playing field: tanks, bases and flags.
3. All games last for a total of ten minutes.
4. Two teams participate in the game, known as blue side and red side.
5. The team with the most number of flag captures at the end of the time period wins the game.
6. Each team consists of a home base, a team flag, and two tanks.
7. A point is scored by a tank capturing an enemy flag and bringing it to the home base. The tank's team flag must also be at the base for a point to be scored.
8. Each tank takes on one of two roles, either attacker or defender. It is the attacker's job to capture the enemy's flag, while it is the defender's job to prevent the enemy from capturing the team's flag.

9. Each tank has a projectile weapon (gun) that can fire one projectile at a time. The projectile travels in a straight line and explodes if it is within five metres of another tank or hits the edge of the playing field. An explosion has a blast radius of ten metres. Any tank within the vicinity of the blast radius is killed and removed from the game for a total of one minute. After one minute the dead tank re-spawns behind its own base and is brought back into the game.
10. Each tank has a sensor with a variable range and field of view. The sensor is used to perceive and identify other tanks, flags and bases.
11. If a tank is carrying a flag while it is destroyed the flag will be dropped at the tank's last position. This allows friendly tanks to rescue the flag automatically returning it to its home base, or enemy flags to seize it and carry it to their home base in attempt to capture it and score a point.
12. Once a flag has been captured, the score for the team that made the capture is incremented and the flag is returned to its home base.

6.4 Architectural Overview

6.4.1 Major Subsystems

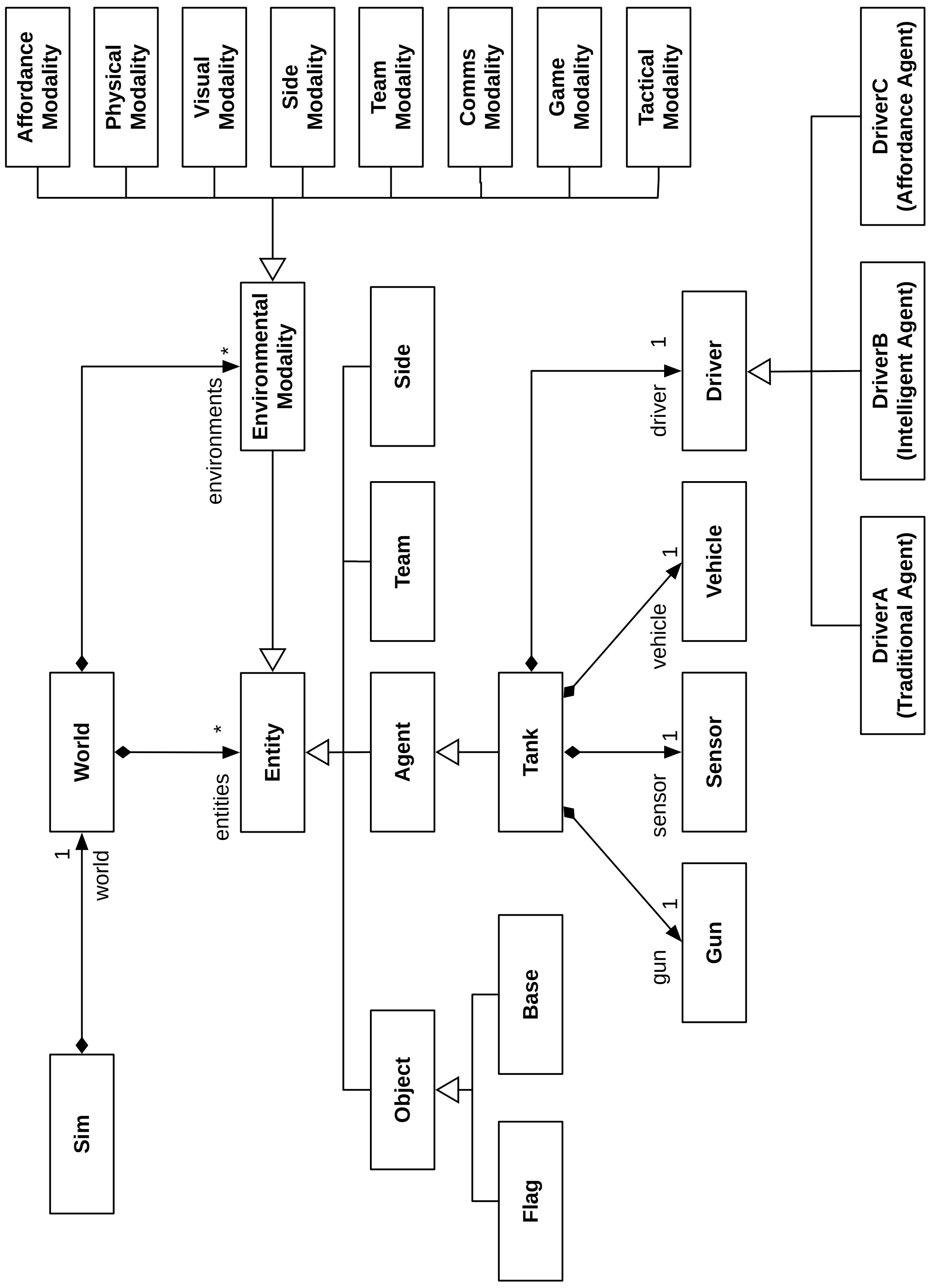
The game of Capture The Flag (CTF) was implemented as a time-stepped simulation using the Python programming language. The tick rate for the simulation was set to one-tenth of a second. The software architecture included explicit computational representations of the world, tanks (game players), sides (red and blue), bases and flags.

The virtual world was implemented as a multi-modal environment, meaning that there were explicit and separate implementations of the visual, physical, social (team and sides), tactical, game and affordance environments. This is using the multi-modal approach to environmental representation described in Chapter 4 and also implemented in HAVE as described in Chapter 2. The exact environmental modalities used in the CTF game are described in more detail in Section 6.4.3. The tank entity was the most complex model in the game and was driven by an intelligent agent representing the driver and is described in more detail in Section 6.4.2.

Figure 6.2 is a UML class diagram depicting the most important classes and the key relationship between them in the design of the CTF multi-agent simulation.

The *Sim* class controls the entire simulation including undertaking the time step execution of the entire simulation. The *Sim* class contains an instance of the entire virtual world of the CTF game which is represented by the *World* class. The *Sim* class instantiates the virtual world dynamically at simulation initialisation time via a file based game specification mechanism. This allows the configuration of the entire simulation to be easily changed.

The virtual world represented by the *World* class consists of a list of entities and a list of environmental modalities which each reference specific entities as needed. Each modality represents a different aspect of the virtual environment in which the game of CTF is being played.



Figures 6.2: Key relationships between primary classes in the CTF multi-agent simulation.

The modalities range from a physical representation of the environment to more abstract environmental representations that represent social or organisational structure in the game, as well as a modality representing the environment from the perspective of the affordances available to agents. The environmental modalities in the CTF multi-agent simulation are described in more detail in Section 6.4.3.

A list of entities come from a set of classes representing all the basic elements present in a game of CTF. These include objects such as flags and bases represented by the *Flag* and *Base* classes respectively.

A tank represents the only type of entity that is also an agent. This is because the tank is the only entity in the simulation which is designed to exhibit pro-active, situated and intelligent behaviour. Sides and teams also represent social and organisational first class entities which can be included in the list of entities. Relationships between different entities are captured in the relevant environmental modality.

Each tank consists of four components. These are a model of the vehicle, a sensor, a gun and a driver agent modelling the cognitive decision making processes of tank driver. Section 6.4.2 describes these components in more detail.

Entities which are also agents play a special role in the simulation. This is not only because they are pro-active and autonomous but also because they are the only entities in which can perceive affordances with respect to other entities in the virtual world.

The software architecture for the CTF simulation consisted of five major subsystems. That is, the major classes and modules shown in the UML diagram in Figure 6.2 (as well as other less important classes and modules not shown in the diagram) belong to one of five major software subsystems making up the CTF software system.

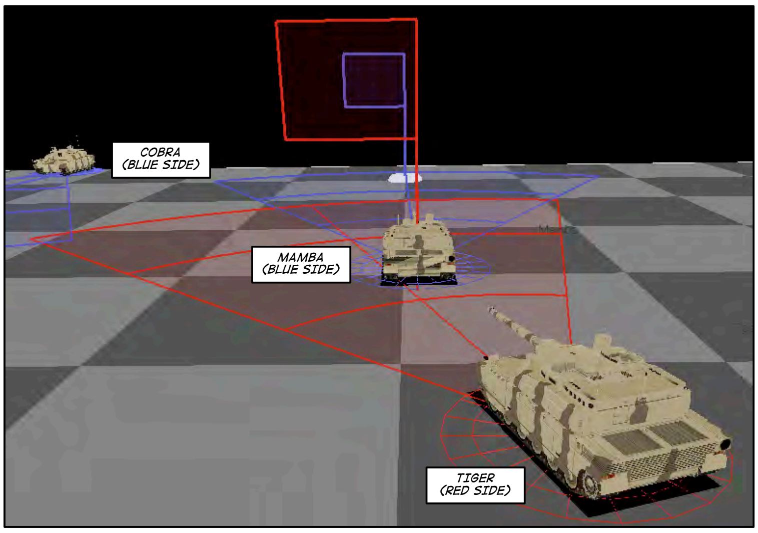
The division into these five major subsystems will allow the empirical evaluation to show how much each subsystem contributes to the overall computational footprint of the simulation. The five major subsystems are as follows.

(1) Simulation: Modules, classes and functions related to time-stepped simulation execution and management of data exchange between the various components in the simulation.

(2) Utilities: Modules, classes and functions providing general purpose utilities to the rest of the simulation architecture, including mathematical functions, modules for logging data and loading files.

(3) Environment: Modules, classes and functions associated with modelling the virtual environment and all its associated modalities. Each modality can be loaded at run-time as needed. This also applies to the affordance modality. When affordances are calculated in the environment this modality is included in the simulation, whereas it is removed when agents calculate their own affordances.

(4) Agent: Modules, classes and functions related to modelling the tank driver agent. This subsystem contains three separate tank driver agents as indicated in Figure 6.2 – DriverA, DriverB and DriverC.



Figures 6.3: Tank Interception. In this screen capture from the visualisation of the CTF multi-agent simulation three tanks are visible. A blue tank (call-sign Cobra) is patrolling as a defender in the distance in the left of the screen capture. In the centre of the figure, another blue tank (call-sign Mamba) has seized the red flag and is on its way to the blue base (where the blue flag is located) in order to capture the flag and score a point for the blue side. However, Mamba has been detected by a red tank (call-sign Tiger) undertaking the defensive role, and attempts to intercept Mamba in the hope of destroying the enemy tank and subsequently rescuing the red flag, so that it can be safely returned to red base. The wedge emanating from each tank is a visualisation of each tanks sensor coverage. The length of the wedge indicates the sensor's detection range, while the width indicates the sensor's field of view. Any entity that appears within this sensor coverage will be automatically detected and tracked by the sensing tank.

- **DriverA** implements a traditional (non-affordance based) agent tank driver using a finite state machine based approach.
- **DriverB** implements an affordance based agent in which the affordances are determined in the agent.
- **DriverC** implements an affordance based agent in which affordances are directly perceived from the environment. The agent considers the affordances available to it and decides which one to adopt.

(5) Entity: Modules, classes and functions associated with non agent entities (such as flags, bases, sides) as well as physical models such as sensor, vehicle and gun models.

6.4.2 Tank and Agent Design

The tank entity was the most complex model in the game and was driven by an intelligent agent representing the driver. Each tank was computationally represented using a number of related simulation models:

Vehicle Model: A vehicle dynamics model that provides each tank with the ability to move around the playing field. The vehicle model is controlled by the driver agent via commands to set speed and heading.

Sensor Model: The sensor model allows the driver agent to perceive the other entities in the world. The configuration of the sensor (its field of view and sensing range) can be changed for each agent. For each entity within the sensor's range and field of view the sensor model generates a track (with information about the perceived entity's position, speed, orientation and type) which are sent to the agent. Visualisation of the tank's sensor coverage can be seen in the visualisation of the CTF simulation in Figure 6.3. The set of fields of view (narrow, medium and wide) and ranges (short, medium, long) used for the sensor model are illustrated in Figure 6.8.

Gun Model: The gun model allows the driver to fire a single projectile at any one time for the purposes of destroying enemy tanks. The projectile does not distinguish between enemy and friendly tanks. It uses a proximity fuse to detonate and destroy any tanks within the specified blast radius (in the simulations described in this Chapter the blast radius was set to 10 metres).

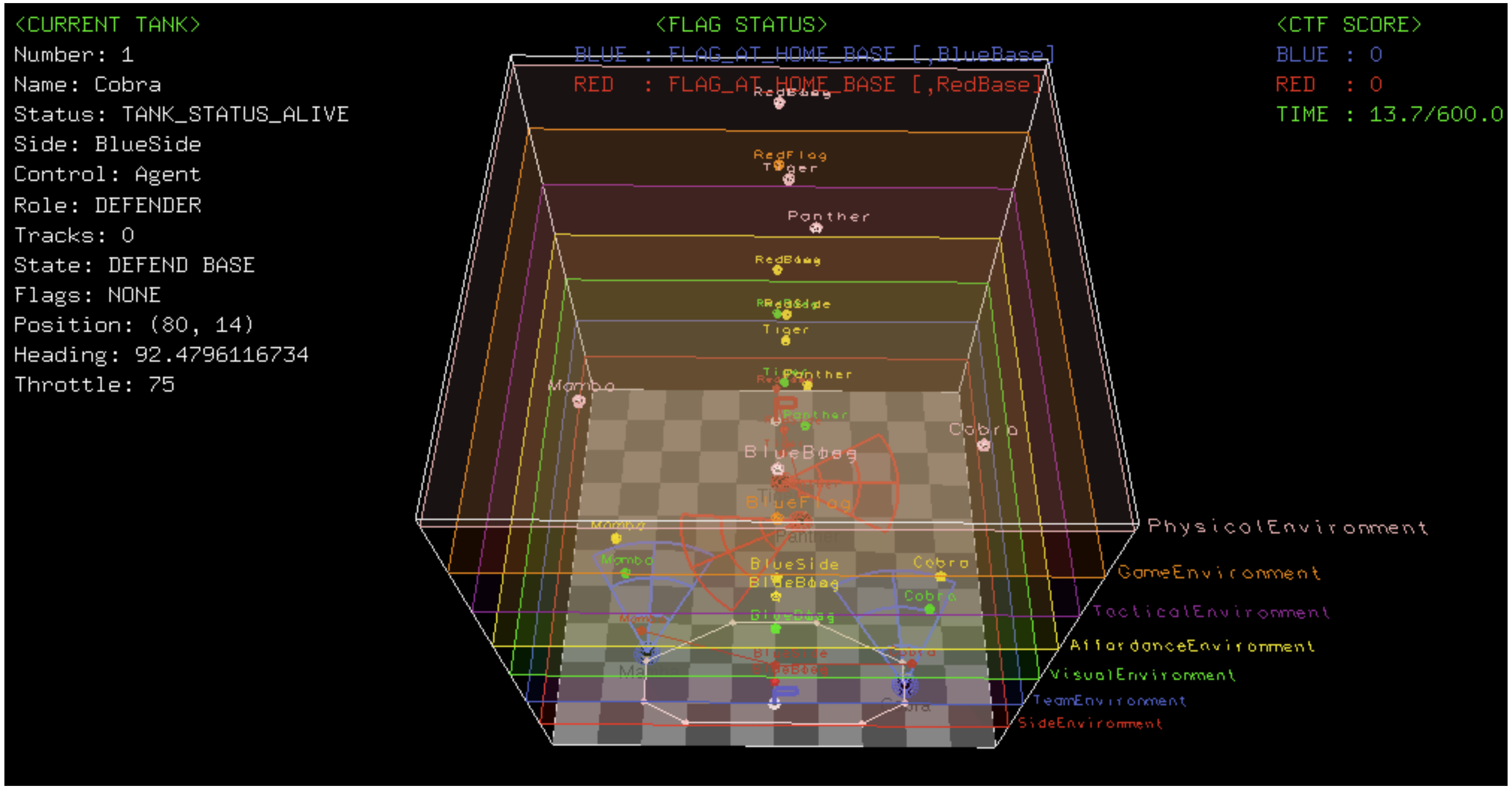
Driver Model: The driver model implements an intelligent rational agent that uses the track information from the sensor model, reasons about what to do and then performs actions in the world by sending commands to the vehicle, sensor and gun models. The driver agent undertakes one of two roles (defender or attacker) and has capabilities that include defending and attacking bases, intercepting enemy tanks, as well as capturing, seizing and rescuing flags. As mentioned in Section 6.4.1 three types of driver agents were implemented. The first, **DriverA**, uses a traditional agent based approach making use of a finite state machine to represent the different states of the driver agent during a game. The second, **DriverB**, uses an affordance based approach where the affordances are determined by each agent. The third, **DriverC**, directly perceives affordances from the environment and is only responsible for deciding which affordance to adopt. Details on the affordance based approach used here can be found in Section 6.5.

Tank Model: The tank model aggregated the vehicle, sensor, gun and driver models and provided a mechanism for data exchange between them. The tank model also acted as an interface to the rest of the environmental model.

Figure 6.3 shows the 3D visualisation of CTF multi-agent simulation game instance. In this particular screenshot, one tank can be seen intercepting another. The visualisation includes representations of the game map, the tanks, the flags on each side and the sensor coverage for each tank.

6.4.3 Environmental Design

Each environmental modality was implemented using a directed acyclic graph (DAG). The nodes in each graph represent the manifestation of specific entities in a particular environmental modality, while the arcs in the graph represent a specific type of relationship relevant to the particular environmental modality.



```
<CURRENT TANK>
Number: 1
Name: Cobra
Status: TANK_STATUS_ALIVE
Side: BlueSide
Control: Agent
Role: DEFENDER
Tracks: 0
State: DEFEND BASE
Flags: NONE
Position: (80, 14)
Heading: 92.4796116734
Throttle: 75
```

```
<FLAG STATUS>
BLUE : FLAG_AT_HOME_BASE [,BlueBase]
RED  : FLAG_AT_HOME_BASE [,RedBase]
```

```
<CTF SCORE>
BLUE : 0
RED  : 0
TIME : 13.7/600.0
```

Figures 6.4: The visualisation software for the CTF multi-agent simulation allows for the independent visualisation of all the environmental modalities. Each environmental modality is represented as a 2D plane overlaid over the game map in a different colour. Each plane renders all entities that manifest themselves in a particular modality. Additionally the underlying graph structure for each modality is also rendered indicating the relationships appropriate for a particular environmental modality. In the case of the affordance environmental modality (yellow plane) all the entities are represented as nodes and affordances between entities are represented as labelled directional arcs.

Figure 6.4 shows an abstract visualisation of the different environmental modalities represented in the CTF multi-agent simulation. A brief overview of the role each modality played in the simulation follows.

6.4.3.1 Physical Environmental Modality

The purpose of this environmental modality was to represent the physical entities and their physical interactions in the CTF multi-agent simulation. All the entities in the simulation that had physical manifestation (such as tanks, flags and bases) were represented in this modality.

The main interaction that was modelled and represented was collisions and collision response. Three types of collisions were modelled. Collisions between tanks and the bounds of the game map, collisions between exploding projectiles fired by a tanks guns, and collisions between tanks. A collision between an exploding projectile and a tank causes the tank to be destroyed. This is counted as a kill and the tank is subsequently removed from the game map.

6.4.3.2 Side Environmental Modality

The side modality captures the relationships between the various entities in the game and the side they are playing on. While in this game blue and red sides are represented, the implementation is generic enough to handle an arbitrary number of sides.

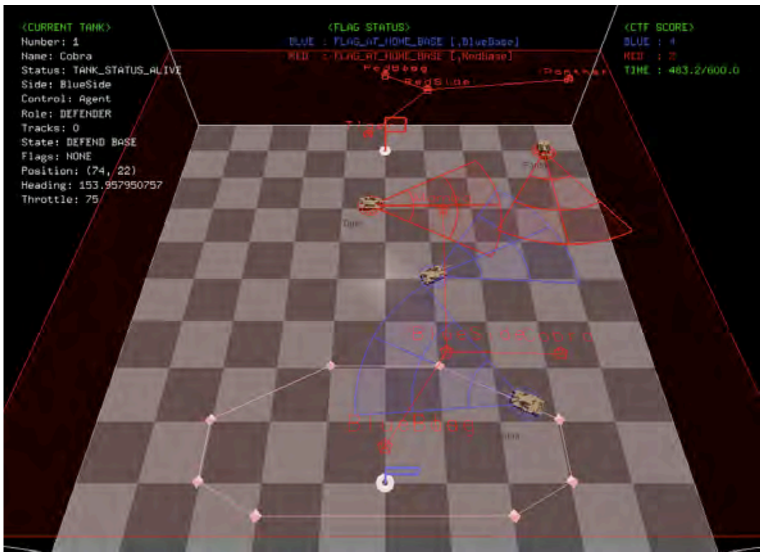
In addition to entities such as flags, bases and tanks which manifest themselves in the physical environment, the concept of a side is treated as a first class entity in this simulation. It has no physical manifestation. Rather, it is an abstract social construct that is explicitly represented in this modality.

The nodes in this graph therefore do not only contain the various entities such as tanks, flags and bases, but a node also exists for each side in the game. The directed arcs in the graph represents the relationship of belonging to a particular side, in this case being either blue side or red side. Figure 6.5 shows a visualisation of the side modality from a game of CTF.

6.4.3.3 Team Environmental Modality

The team environmental modality was an extension of the side modality. While sides represent the highest level of team structure in a game of CTF, it is possible for a side to be composed of many sub-teams that are role based. For example, a side may consist of attacking, defensive and scouting sub-teams each one made up of one or more tanks. In the CTF games used in the experiments presented in this chapter, each side consisted of two teams, an attacking and a defensive team, each with one tank (total two tanks per side).

While the concepts of side and team represent similar social relationships in virtual environment, the side environmental modality was deliberately separated from the team modality due to the special role that sides play in the domain of capture the flag.



```
<CURRENT TANK>
Number: 1
Name: Cobra
Status: TANK_STATUS_ALIVE
Side: BlueSide
Control: Agent
Role: DEFENDER
Tracks: 0
State: DEFEND BASE
Flags: NONE
Position: (74, 22)
Heading: 153.957950757
Throttle: 75

<FLAG STATUS>
BLUE : FLAG_AT_HOME_BASE [,BlueBase]
RED  : FLAG_AT_HOME_BASE [,RedBase]

<CTF SCORE>
BLUE : 4
RED  : 2
TIME : 483.2/600.0
```

Figures 6.5: A screen capture of the CTF multi-agent simulation indicating the game map with the side environmental modality shown as the red 2D plane layered on top of the game map. This modality shows the two side graphs (one for red side and one for blue side). Abstract non-physical but social entities such blue side and red side which do not appear on the game map can be seen in the side modality. The positions of these entities are taken as the average of all other entities belonging to each side.

6.4.3.4 Visual Environmental Modality

The visual environmental modality was used to represent entities that can be visually detected by each tank's sensors. This corresponds to physical entities such as tanks, flags and bases and did not include abstract entities such as teams and sides. Each entity in this modality is represented as a node in the graph. A directed arc between nodes indicated whether one entity was visible to another.

6.4.3.5 Communications Environmental Modality

The communications environmental modality is used to represent the communications environment. Entities that can communicate (in this case only tanks) are represented as nodes in a graph. Lines of communication are represented as directed arcs between nodes. In most cases all entities on the same side can communicate with each other. However, the representation allows for more sophisticated communication relationships. For example, it allows communication to be restricted to within sub-teams to enable complex coordination.

6.4.3.6 Tactical Environmental Modality

The tactical environmental modality allows for the representation of information in the environment that is tactically significant to the players of the CTF game. This typically did not include entities that manifest themselves physically in the game (such as tanks); rather the modality is used to represent aspects of the environment which provided meaningful tactical information to the various players.

The tactical modality also includes information such as way-points used by blue or red tanks. The way-points are typically part of more complex tactical representations of the virtual environment such as defensive patrols and attack patterns.

6.4.3.7 Game Environmental Modality

The game environmental modality is used to represent aspects of the virtual environment which captured information about the overall game of CTF being played. This includes determining if a flag had been captured (and adjusting the overall score accordingly), as well as maintaining various information about the progress of the game such as which tanks are alive and which are dead, the elapsed time since a tank was killed (to allow for re-spawning), management of re-spawning, kill records, overall score as well as keeping track of the overall game time.

6.5 Affordances in the Capture The Flag Simulation

6.5.1 Affordance Environmental Modality

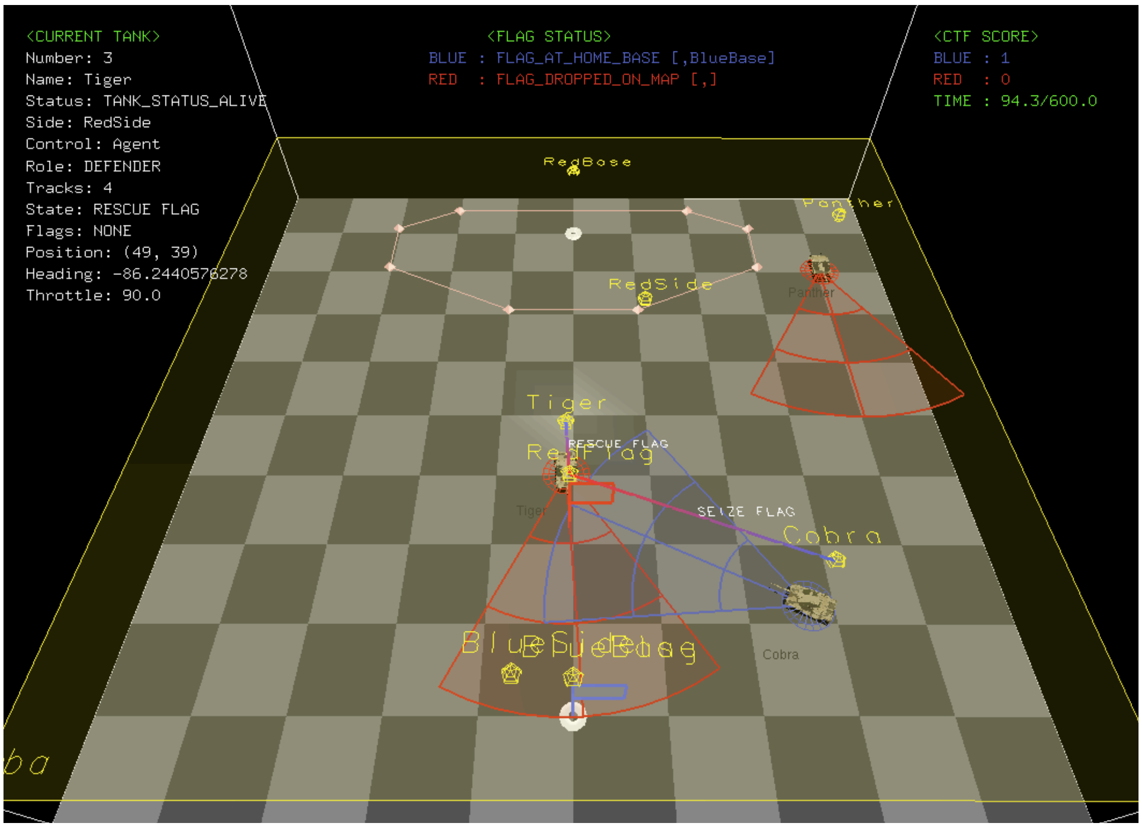
The affordance environmental modality in the CTF multi-agent simulation is the aspect of the virtual environment responsible for representing the affordances present in the world and responsible for computing these affordances. In the cases where each agent determined its own affordances, the affordances environmental modality was not used in the multi-agent simulation.

However, in the cases where affordances can be directly perceivable in the environment, it is this subsystem which is responsible for representing and computing all of these affordances. Just like any other simulation component this modality can be considered as a time stepped model.

Each simulation time step the affordance modality is updated. It takes into account information about each individual agent and information from each of the other environmental modalities into consideration in making an assessment of whether an affordance was available to an agent.

In this simulation the list of all possible affordances are pre-defined, and it is a matter of determining which ones are applicable at any given time step for each agent in the simulation. In this way the affordance modality acts as a conduit between the agents and the environment, which in this case is represented by all the other environmental modalities.

This modality captures the concept from ecological psychology that although affordances are directly perceivable in the environment, they do not exist in the agent or in the environment, rather they are relation that arise between the two.



```
<CURRENT TANK>
Number: 3
Name: Tiger
Status: TANK_STATUS_ALIVE
Side: RedSide
Control: Agent
Role: DEFENDER
Tracks: 4
State: RESCUE FLAG
Flags: NONE
Position: (49, 39)
Heading: -86.2440576278
Throttle: 90.0
```

```
<FLAG STATUS>
BLUE  : FLAG_AT_HOME_BASE [,BlueBase]
RED   : FLAG_DROPPED_ON_MAP [,]
```

```
<CTF SCORE>
BLUE  : 1
RED   : 0
TIME  : 94.3/600.0
```

Figures 6.6: Visualisation of the affordance environmental modality is shown here as the yellow 2D plane overlaying the game map. In this type of visualisation affordances are represented as directional labelled associations or relationships between agents and the entities that afford a particular action. In this screen capture from the CTF multi-agent simulation, two affordance relationships are shown with respect to the red flag. In this case the second blue tank had seized the red flag and was making its way to the blue base to capture the flag and score a point. On its way it was shot by a red tank and the red flag was dropped on the map. A blue tank (call-sign Cobra) and a red tank (call-sign Tiger) both detect the dropped flag. The flag affords a different course of action for each tank. For the red tank, the flag affords being rescued to prevent it being captured by the blue tanks, whereas for the red tank, the flag affords being seized in order to effect a flag capture and score a point. Both these affordance relationships are indicated as directed labelled arcs between Cobra and the red flag and between Tiger and the red flag in the visualisation of the affordance environmental modality.

The affordance relationships in the affordance environmental modality are represented as directed arcs in a directed graph, with the nodes representing entities in the world. Each affordance in the system is represented by a single directed arc between two nodes. Each arc is annotated with the type of the affordance. The source of each arc can only emanate from an entity node that is also an agent. That is, there is a distinction between regular entity nodes and agent nodes in the in the graph representation.

In the case of the CTF game, the only agents in the simulation are the tanks. The act of direct perception of affordances in the environment is reduced to providing each agent a list of arcs emanating from it to other entities in the world, with each arc representing an affordance available to the agent.

6.5.2 Generating Affordances

Affordances can be incorporated into a virtual environment by statically annotating entities in the world with the actions which can be performed on them by agents. These annotations are made by the designers of the multi-agent simulation or are pre-computed before a simulation is run. Agents can perceive these annotations and decide which particular action to undertake.

This approach has a number of limitations. Affordances are dynamic and change over time as opposed to being just static annotations. More importantly, affordances are dependent on the agent observer, the entity being observed, and the relationship between the two, instead of just the properties of the entity. That is, different agents need to perceive different affordance annotations. The variation in the agent can be in the agent's type or even in its physical and mental states and capabilities.

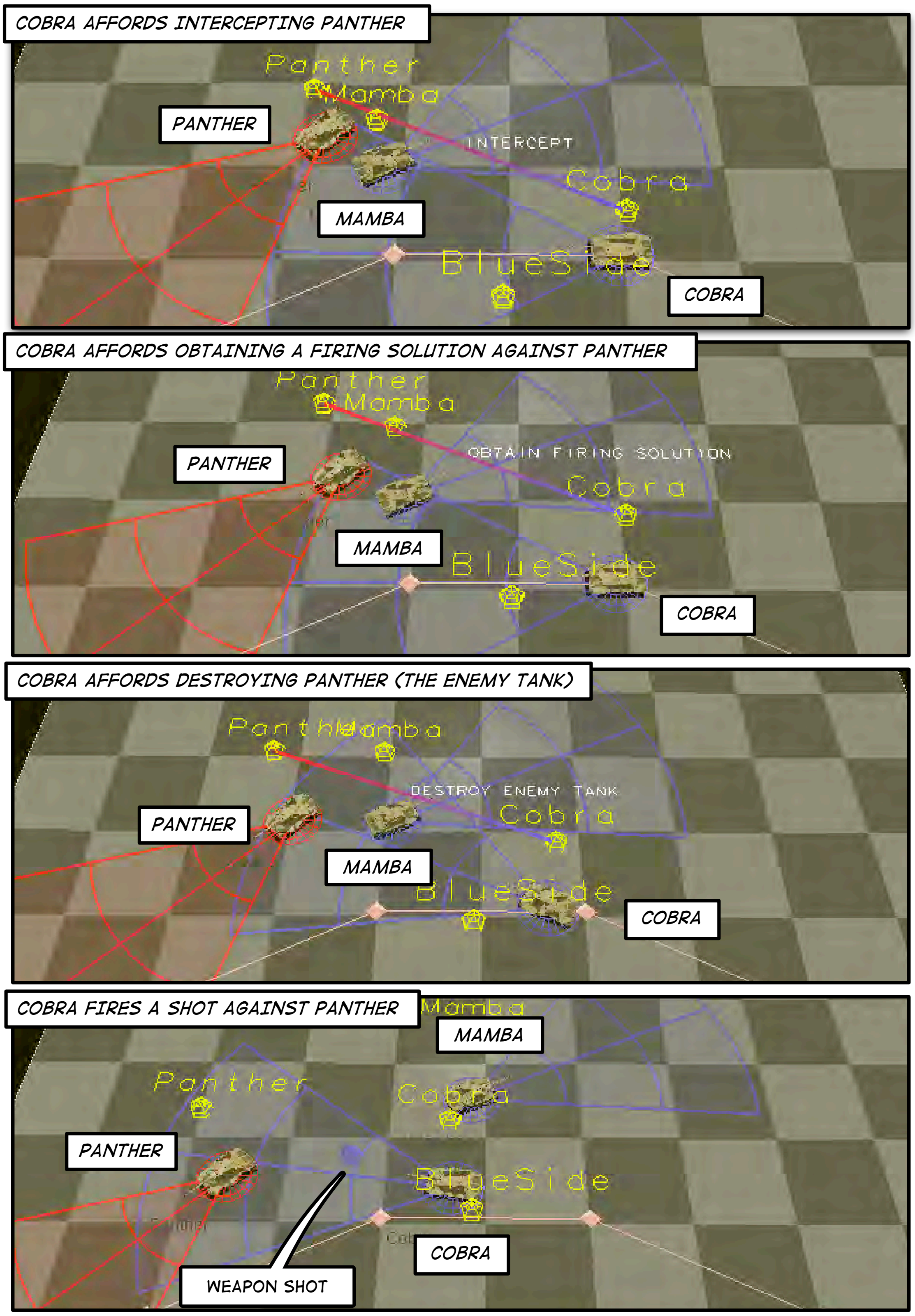
The approach taken here to developing an affordance based approach to agent reasoning is to dynamically compute the affordances for each individual agent at run time. Although this is computationally more expensive than a static annotation of the virtual environment, it provides for a richer and more realistic model of agent-environment interaction.

In order to dynamically compute affordances for an agent perceiving a specific entity at run time, the following factors are taken into consideration.

Entity State The observable properties, status and type of an entity will change the affordances an agent perceives. Is the entity a tank or a flag? Is it a blue tank or a red tank? Is the tank attacking my home base? The answers to these questions change the action possibilities an observing agent will perceive with respect to this entity.

Agent Physical State The agent's physical state will also affect which affordances are available. Does the speed of tank affect its ability to intercept an opponent? How about the sensory range and field of view? If the tank is out of ammunition then the action possibility to destroy another tank becomes non existent.

Agent Physical and Mental Capabilities An entity does not afford a course of action for an agent if that agent does not have the capability to perform the action. An enemy tank does not afford being destroyed if the agent has no gun.



Figures 6.7: Affordance Visualisation - Sequence of Tank Destruction. (i) Cobra is afforded the opportunity to intercept Panther. (ii) Cobra is afforded the opportunity to obtain a firing solution against Panther. (iii) Cobra is afforded the opportunity to destroy the enemy tank Panther. (iv) Cobra fires a shot at Panther in order to destroy it.

Agent Mental State An important determinant of affordance is an agent's mental state. How do mental attitudes such as beliefs, goals and intentions influence the affordances the agent can perceive? For example, a tank agent may only attack enemy tanks in its vicinity if it has an intention to defend the home base.

Agent-Entity Relations Specific agent-entity relationships affect what the entity affords the agent. For example, are the agent and entity on the same team? How far apart are they?

Two options for incorporating this model of affordance into a multi-agent simulation are explored. In the first case, the affordances are dynamically computed in the environment. This is a dynamic extension of the virtual environment where affordances have been added to entities as static annotations. From one perspective the virtual environment in the CTF multi-agent simulation can be considered an affordance based self-annotating virtual environment.

The second case allows for each individual agent to dynamically compute its own affordances with respect to entities it can perceive in the world. Although this second case is not in the spirit of the direct perception theory of affordance, it is consistent with traditional approaches to agent reasoning where the predominance of intelligent reasoning occurs in an agent.

This has an important implication for the design of the agent-environment interface and raises the question of where in the simulation system should affordances be computed or determined. Should the computation occur in the agent, as is traditionally the case for determining courses of action, or should they be determined in the environment (or the agent-environment interface) as suggested by ecological psychology?

6.5.3 Affordance Generation Algorithm

In this Section the high level algorithm which generates the affordance relation for each agent with every other entity in the environment is described.

For every agent in the environment, the algorithm considers the pair-wise relation with every other entity in the environment including other agents. For each agent-entity relation, a list of possible affordances from a library of affordances are evaluated. If the affordance exists, it is added to the set of affordances for that agent indexed by the current time and the entity to which the affordance applies. Therefore, at each time step the algorithm returns a set of affordances indexed by agent, entity and current time.

The general nature of the algorithm evaluates all possible affordances from the affordance library for each agent-entity interaction. If the affordance library is large, this becomes potentially computationally expensive. A special case variant of this algorithm evaluates only affordances from the library which are relevant to the agent-entity combination under consideration. This reduces the number of affordance evaluations.

The description above assumes the affordance generation algorithm is located in the environment as it has access to all global information. However, an alternative is to modify the algorithm and allow each agent itself the affordances which are relevant to it, for the entities it can perceive in the world using its own internal beliefs.

Algorithm 6.1 Affordance Generation in Capture The Flag

```
1: procedure GenAffordances($agents$, $entities$, $\Phi$, $t$)
2:     affordances($\alpha, \varepsilon, t$) $\leftarrow \{\}$
3:     for all $\alpha \in agents$ do
4:         for all $\varepsilon \in entities$ do
5:             if $\varepsilon \neq \alpha$ then
6:                 $\Phi' =$ relevant($\alpha, \varepsilon, \Phi$)
7:                 for all $\phi \in \Phi'$ do
8:                     if affords($\alpha, \varepsilon, \phi$) $= true$ then
9:                         affordances($\alpha, \varepsilon, t$) $\leftarrow \phi$
10:                    end if
11:                end for
12:            end if
13:        end for
14:    end for
15:    return affordances
16: end procedure
```

This gives us four variations on the one algorithm. A general and specific version in the environment, and a general and specific version in the agent. This is discussed in more detail in the context of agent-environment architectures in Section 6.6.

A pseudo-code outline of the affordance algorithm is presented in Algorithm 6.1. The following notation is used:

- The set of all agents in the world is given by *agents*, the type of an individual agent being *Agent*, and α used to index or represent an individual agent.
- The set of all entities in the world (including agents) is given by *entities*, the type of an individual entity being *Entity*, and ε used to index or represent an individual agent.
- The set of all entities in the world includes all agents such that $agents \subseteq entities$.
- The current time is denoted by t.
- The library of affordances is defined by Φ and the subset of the library which is relevant for a given agent α and entity ε is given by Φ' and is found using the function relevant($\alpha, \varepsilon, \Phi$).
- The set aff(α, ε, t) contains all the possible affordances that agent α could have with respect to entity ε at time t.
- The relationship affords($\alpha, \varepsilon, \phi$) is true for any entity ε which affords ϕ for an agent α, otherwise it is false.

The consideration of only the relevant affordances from the library as indicated by line 6 in Algorithm 6.1 distinguishes the general and the specific versions of this algorithm. For the case when the algorithm is computed inside the agent, the outer for loop (lines 3-14) is omitted and only the inner loop evaluating affordances with respect to each entity is considered.

The output from the affordance generation algorithm is then presented to each agent. Each agent has the possible actions that entities in the environment afford it. It is up to the agent to decide which affordance to adopt (if any) and to carry out the necessary actions.

6.6 Agent-Environment Architecture Types

Five variations of the CTF simulation architecture were developed. Most components in the system remained identical. Only the components dealing with the agent-environment interface were modified. This allows an empirical comparison of the differences in computation time for different approaches to implementing an affordance based agent-environment interaction.

Four affordance based architectures were developed to experimentally compare the performance of the various approaches. In two of these the affordances were computed in the environment:

- (1) Intelligent Environment I (General)
- (2) Intelligent Environment II (Specific)

and in the other two the affordances were computed in the agent:

- (3) Intelligent Agent I (General)
- (4) Intelligent Agent II (Specific)

The possible affordances for each agent are evaluated once every time step. However, two variants of the affordance generation algorithm are considered. The first one is the most general algorithm and considers affordance possibilities for agent with respect to every other entity in the simulation.

An alternative variation involves a less general (more specific) algorithm which takes domain information into account (such as information about sides and entity types) to reduce the number of affordance evaluations, and hence the number of comparisons.

The final variant considered:

- (5) Traditional Agent

uses a finite state machine model of the tank driver agent implemented using a traditional sense-reason-act model of agent reasoning. This was developed for the purposes of testing the architecture and as a simple base line reference case.

Although this agent model could successfully play a game of CTF, it was behaviourally simpler than the other architectural variants and hence it is not necessarily meaningful to directly compare the computation times for this approach with more complex approaches.

This gives a total of four affordance based architectures and one traditional agent architecture as summarised below and in Table 6.1.

(1) **Intelligent Environment I (General)** An affordance based architecture in which the affordances for each agent are computed in the environment. A general affordance generation algorithm is used evaluating all possible affordances for each agent-entity relation. This architecture makes use of the affordance environmental modality in conjunction with the **DriverC** agent driver model.

(2) **Intelligent Environment II (Specific)** An affordance based architecture in which the affordances for each agent are computed in the environment. A specific affordance generation algorithm is used to reduce the number of affordance evaluations for each agent-entity relation. This architecture makes use of the affordance environmental modality in conjunction with the **DriverC** agent driver model.

(3) **Intelligent Agent I (General)** An affordance based architecture in which the affordances for each agent are computed in the agent module. A general affordance generation algorithm is used evaluating all possible affordances for each entity the agent can perceive. This architecture makes use of the **DriverB** agent driver model and does *not* use the affordance environmental modality.

(4) **Intelligent Agent II (Specific)** An affordance based architecture in which the affordances for each agent are computed in the agent module. A specific affordance generation algorithm is used to reduce the number of affordance evaluations for each entity the agent can perceive. This architecture makes use of the **DriverB** agent driver model and does *not* use the affordance environmental modality.

(5) **Traditional Agent** An agent architecture based on the traditional Sense-Reason-Act model of agent reasoning implemented using a finite state machine approach. This architecture makes use of the **DriverA** agent driver model and does not use the affordance environmental modality.

Table 6.1 summarises how affordances are computed for each architecture type under consideration by listing where affordances are computed (environment or agent) and whether a general purpose of specific affordance generation algorithm is used. Table 6.2 summarises the use of different agent tank drivers and the use of the affordance environmental modality for each architecture type under consideration.

Note that at times in this Chapter, the notation for the various architecture types will drop the (General) or (Specific) post-fix for brevity. This will typically be the case on charts.

6.7 Evaluation Criteria

In order to compare the different architecture types a number of evaluation criteria were selected. The criteria fell into two categories. The first category measured information about the outcome of the games of CTF being played in the experiment. This included information such as the total number of flags captured by each side (e.g. the final score of each game), as well as the number of tanks killed in each game. While these metrics were not intended to compare the design of different affordance based approaches per se, they were used to check on the viability of each architecture type by making sure that a reasonable game of CTF was being played.

Tables 6.1: A list of the five agent-environment interaction architectures implemented in the Capture The Flag simulation. The table indicates for each architecture type being considered (Column 1) where affordances are computed (Column 2), and whether a general purpose or specific affordance generation algorithm is used (Column 3).

Architecture Types Implemented		
Architecture Type	Affordances	Affordance Generation
Intelligent Environment I (General)	Environment	General
Intelligent Environment II (Specific)	Environment	Specific
Intelligent Agent I (General)	Agent	General
Intelligent Agent II (Specific)	Agent	Specific
Traditional Agent	n/a	n/a

Tables 6.2: Architecture/Agent/Affordance Modality Combinations. This table indicates the type of driver agent (Column 2) and the use of the affordance environmental modality (Column 3) for each of the five architecture types being considered (Column 1).

Architecture Types Implemented		
Architecture Type	Driver Agent	Affordance Modality
Intelligent Environment I (General)	DriverC	Yes
Intelligent Environment II (Specific)	DriverC	Yes
Intelligent Agent I (General)	DriverB	No
Intelligent Agent II (Specific)	DriverB	No
Traditional Agent	DriverA	No

The second set of criteria attempted to assess each architecture type by measuring the computational footprint of each architecture type. The computational footprint also allowed for an indirect measure of the overall architectural complexity of each architecture type. The measurement was made by using a deterministic profiler.[60]

For each game of CTF the profiler logged a number of parameters for each Python function or method call. These included the number of times a function was called (including an indication of recursive calls), the total time spent in each function (excluding calls to sub-functions), and the cumulative time spent in each function (which included calls to sub-functions). The profiler also produced data regarding the total number of calls and the total overall computation time of the simulation.

All this data was generated in a profiler log file. The size of the file reflected the number of function calls that were logged by the profiler. Therefore, the size of the log file was an indirect measure of the computational footprint of a particular simulation run.

By running a number of different CTF simulation scenarios (e.g. different game scenarios) for each architecture type it was possible to look at the mean value and the spread of the result using the standard deviation for a number of the parameters that were of interest.

Also by processing the profiler log data and categorising each function call by the subsystem it belonged to, it was possible to see how much computational effort was being spent on

[60]Since the CTF simulation was programmed with the Python programming language, the standard Python deterministic profiler called HotShot was used to obtain data on the running of the simulation.

each subsystem (Simulation, Utilities, Environment, Agent and Entity as per Section 6.4.1).

The total computation time, cumulative computation time and total number of function and method calls for a simulation run provides a run time snapshot of the computational footprint of each architecture type being considered. Additionally it allows one to see which subsystems make the largest contribution to the overall computation time for the different affordance based approaches being looked at.

The evaluation criteria were deliberately selected to reflect the run time complexity of each architecture type and do not measure its static complexity. For example, the metrics chosen do not provide any insight into how much time it would take a programmer to code an affordance architecture of particular type. This is deliberate because static software engineering metrics (such as lines of code, cyclomatic complexity and function point analysis) are not suited to providing the type of information regarding these architectures that this Chapter aims to evaluate. There are a number of reasons for this.

First, in application domains such as military operations research computational performance is an extremely important non-functional requirement when developing a multi-agent simulation. Hence it is important to ensure that the introduction of an affordance based approach does not impact on the simulation run time in a significantly negative manner and that the simulation can still execute many times faster than real time for Monte-Carlo batch processing.

Second, a static analysis of the CTF simulation architectures are likely to provide a single misleading and not very informative data point. This is because the impact on static software engineering metrics (lines of code, number of classes etc) will greatly depend on the application domain at hand and the specific functional and non-functional requirements of the multi-agent simulation in question. It difficult to extrapolate to other affordance based systems with only a single application domain (in this case CTF).

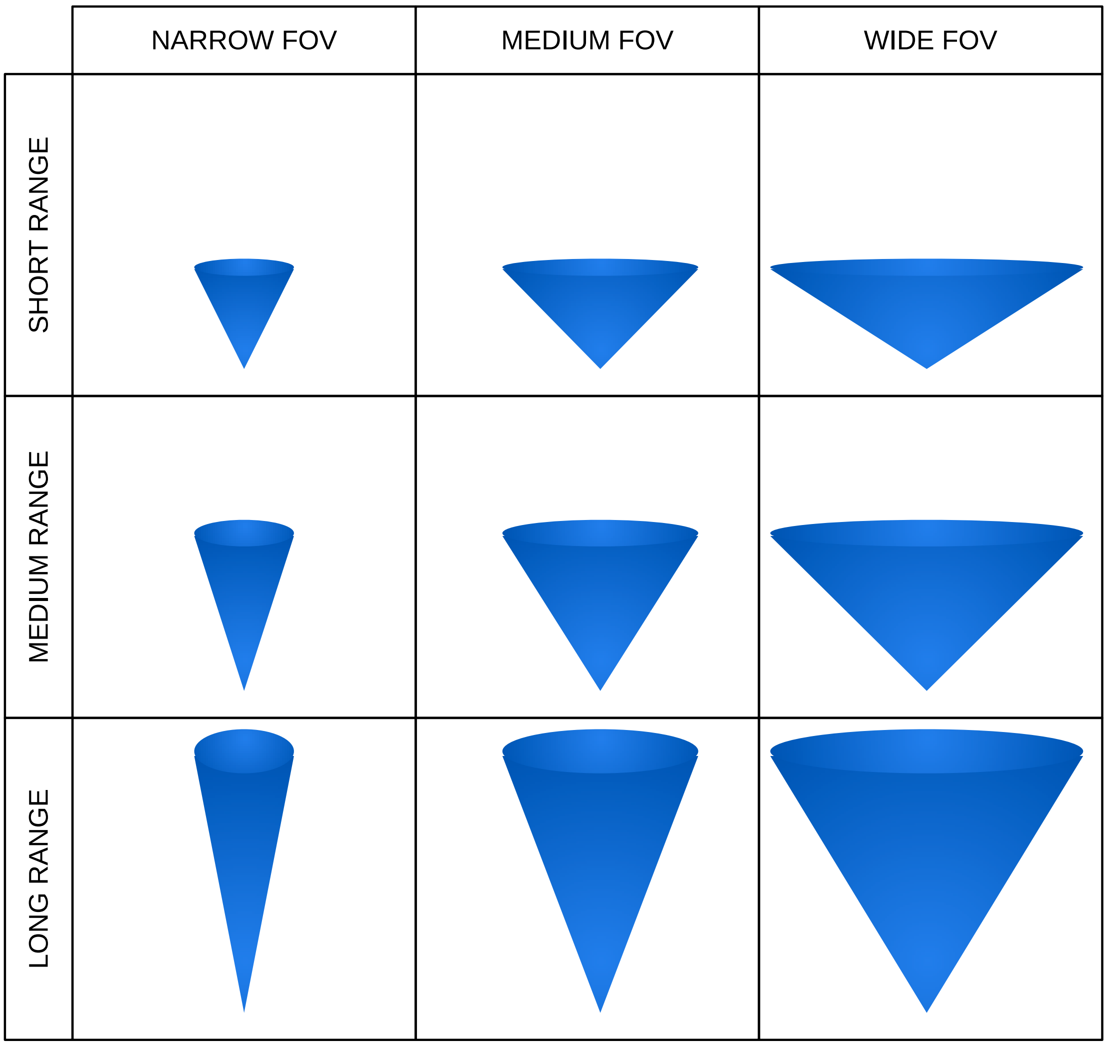
6.8 Experimental Setup

In order to explore a range of agent behaviours and gain statistically significant results, a number of important parameters were varied in the experiment. These included varying the tank's speed, the range and field of view of the sensor, as well as the attack patterns driven by the tanks in the attacking role. Table 6.3 shows the values selected for each parameter of interest.

A sweep of all the parameters was conducted resulting in a total of 108 possible configurations.[61] When this is multiplied by the number of architecture types of interest (5), it results in 540 possible starting configurations for the game of CTF. Therefore, the experiment involved running 540 different versions of a 2v2 CTF game, each running for a total simulated game time of 10 minutes. This is the equivalent of playing 5400 minutes (or 3.75 days) of CTF.

The code was instrumented to output relevant data at the end of each game simulation. This included information such as the number of flag captures per side and the number of

[61]This number was obtained by multiplying the number of values for each parameter from Table 6.3. So that $3 \times 3 \times 3 \times 1 \times 4 = 108$.



Figures 6.8: Sensor Parameterisation. The ability of the tank's sensor to detect and track other entities in the world was determined by two parameters - the sensor's range and field of view. Each parameter was evaluated to three different values. The sensor range was set to a short, medium and long range, while the field of view was set to a narrow, medium and wide. In total this gave nine possible sensor configurations which could be explored. The sensor's ability to detect other entities in the virtual world dramatically impacted on the outcome of a CTF game.

Tables 6.3: The list of the experimental parameters used in the simulation runs. The parameters were selected to provide a broad set of behaviours and statistical variation in the results. By considering all possible combinations of parameters a total of 108 starting configurations are considered for each architecture type. Figure 6.8 graphically illustrate the different sensor parameter configurations used in the experiments. For tank driver agents undertaking the defensive role a single defensive patrol pattern was used for both red and blue side. This involved patrolling using a elliptical defence perimeter around the center of the home base. For tanks in the attacking role two attack patterns were used (2 for red side and 2 for blue). One was starting out to the left of home base and attacking the enemy base from the right, and the other was starting on the left hand side and attacking the enemy base from the left.

Experimental Parameters	
Parameter	**Values (#Parameters)**
Tank Speed	Slow, Medium, Fast (3)
Sensor Range	Short, Medium, Long (3)
Sensor Field of View	Narrow, Medium, Wide (3)
Defensive Patrol	Center (1)
Attack Pattern	Red Left To Right, Red Right To Left Blue Left To Right, Blue Right To Left (4)

tanks killed per side. However, the computational performance of the various architectures was of more interest, and hence the experiment involved executing the simulations using a deterministic code profiler. This provided information such as the cumulative time spent, and the number of method calls to each module.

Therefore, by post processing the results provided by the instrumented simulation and from the profiler, a picture of the computational footprint of the different architecture types as function of the different subsystems began to emerge.

Given that each architecture was executed in 108 different game scenarios, the exercising of the parameter space allowed for the relevant variables to be presented as statistical means with a confidence interval indicated by either the standard error or the standard deviation.

6.9 Discussion of Results

The experiments produced results falling into a number of categories. The first set of results can be categorised as Measures of Effectiveness or MOEs. These results provide information about the outcome of the many CTF games played in the experiment and were obtained by instrumenting the multi-agent simulation to record the required parameters. They are primarily concerned with measures such as the number of flag captures on each side (a measure of which side won the game) as well as other measures which provide an insight into how games progressed, such as the number of tanks killed.

In an operations research context it is these results that matter and are used to advise on the effectiveness of one platform or tactic versus another. In the context of this research these types of results are of secondary importance. However, they are included because they provide relevant background and ground the multi-agent simulation in an actual application domain.

The results obtained from the profiling of the simulation runs provide a number of measures of architectural and computational complexity. These results allow for a computational run time comparison of the different agent-environment architecture types. While a number of software engineering metrics exist that allow for the static analysis of a design architecture, the focus here is on comparing complexity at run time.

As previously described for each architecture type, 108 variations of a 2v2 CTF game were played. Given that five architecture types were considered, a total of 540 games, each of a total of 10 minutes were played. For each architecture type, the mean value of various measures was considered. This allowed for a comparison between architecture types. Looking at the standard deviation and the standard error of a particular measure within an architecture type gave an indication as to how spread out the results were.

6.9.1 Game Measures of Effectiveness

While there are a number of possible ways of measuring the outcome of a capture the flag game, the two most important ones are the number of flag captures and the number of tanks killed during a game. For both these MOEs, it is expected that there is a significant variation between the Traditional Agent architecture and the affordance based architectures.

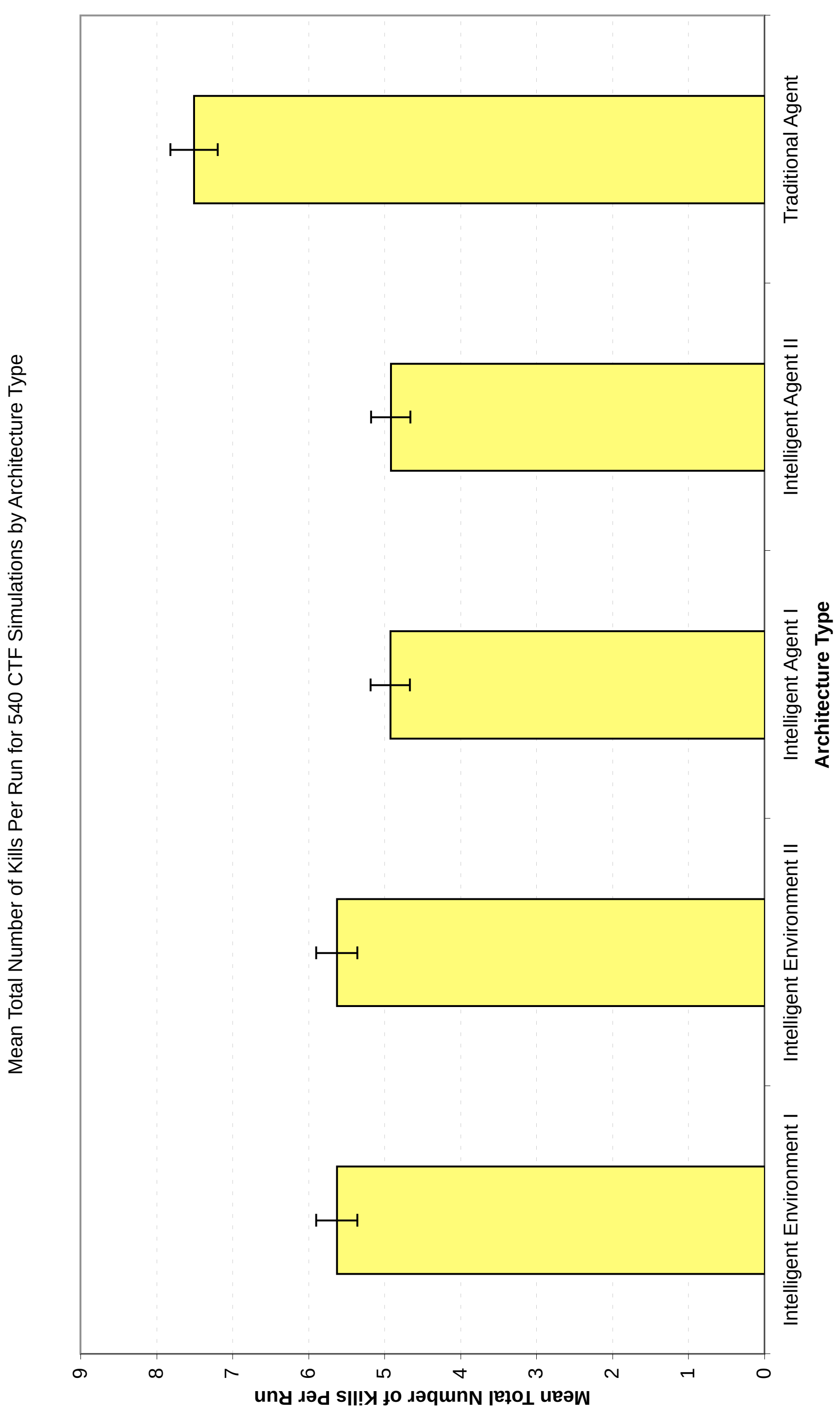
This is due both to the fact that an affordance based approach is used and due to the simpler set of behaviours programmed for the agent. However, it is expected that there should not be significant variation in results between the two Intelligent Environment architectures, and likewise between the two Intelligent Agent architectures.

However, there is expected to be some variation between the two groups of affordance based architectures. While the tanks in both the Intelligent Environment and Intelligent Agent used the same repertoire of behaviours, the variations in how each agent interacted and perceived the environment was enough to affect the outcome of the game.

Chart 1 (Figure 6.9)**.** ***Mean Number of Tank Kills*** *This chart shows the mean number of tanks killed per game of CTF, plotted by architecture type.*

Figure 6.9 plots the mean number of tank kills per ten minute CTF game for each architecture type. This is a mean of the total tank kills, including both blue and red side. The error bars on the chart indicate the standard deviation in the number of kills per game. As can be seen from the chart, there are approximately between seven and eight tanks killed in a ten minute game of CTF when using the Traditional Agent architecture.

For the Intelligent Agent architectures the mean number of tank kills is approximately five, while for the Intelligent Environment architectures the mean number of tank kills is between five and six.



Figures 6.9: Mean Number of Tank Kills by Architecture Type.

Chart 2 (Figure 6.10). ***Mean Number of Flag Captures by Team*** *This chart shows the mean number of flags captured by each side plotted by architecture type. Each flag capture corresponds to a point in a game of CTF. The team with the highest number of flag captures at the end of the game is considered the winner.*

Chart 3 (Figure 6.11). ***Mean Number of Total Flag Captures*** *This chart shows the mean number of total flags capture by architecture type independent of team. It is shows the same information as in Figure 6.10 except the total number of flag captures for the blue and red team have been aggregated.*

Figures 6.10 and 6.11 chart the mean number of flag captures for each architecture type. Figure 6.10 splits up the results and indicates the mean score for both blue and red side, while Figure 6.11 is an aggregate for the mean number of total flag captures within a game. As in Figure 6.9 the error bars indicate the standard deviation in the results from the mean. In all cases the results of the number of flag captures are typical of what one might expect in a ten minute game. [62]

Figure 6.11 shows that when using the Traditional Agent architecture, a flag was captured on average between two and three times a game, while in the affordance based architectures this is slightly higher with a flag capture occurring between three and four times a game.

It is also worth noting that despite the differences between the Intelligent Agent and Intelligent Environment architectures the mean number of total flag captures per game is close enough for the standard deviation error bars to overlap. Due to the differences in the two affordance based architectures it cannot be expected that the fine grained MOEs are comparable, but higher level coarser grained MOEs such as the mean total flag captures are significantly close to each other.

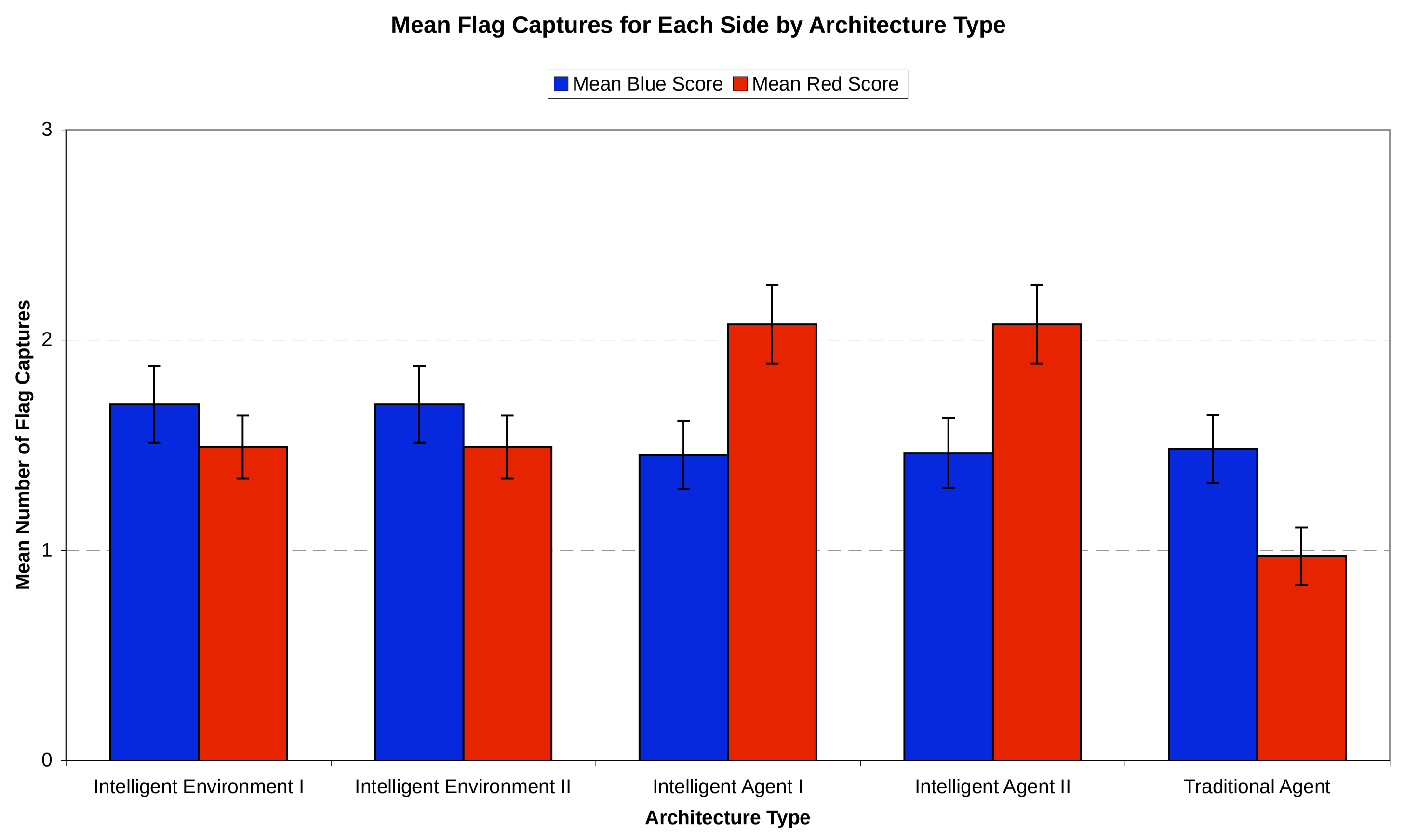
While measures of effectiveness such as tank kills and flag captures that are related to the outcome of the game are not necessarily the most important metrics when comparing affordance based architectures, they do provide an important baseline. What is most important here is not the difference in outcomes between the Intelligent Environment, Intelligent Agent, and Traditional Agent architectures, but the similarities.

It is important to recognise the consistent results between architectures Intelligent Environment I (General)and Intelligent Environment II (Specific) and Intelligent Agent I (General) and Intelligent Agent II (Specific). This consistency is shown across Figures 6.9, 6.10 and 6.11. The consistent game outcomes will be contrasted in the following sections with differing outcomes in other metrics such as computational performance.

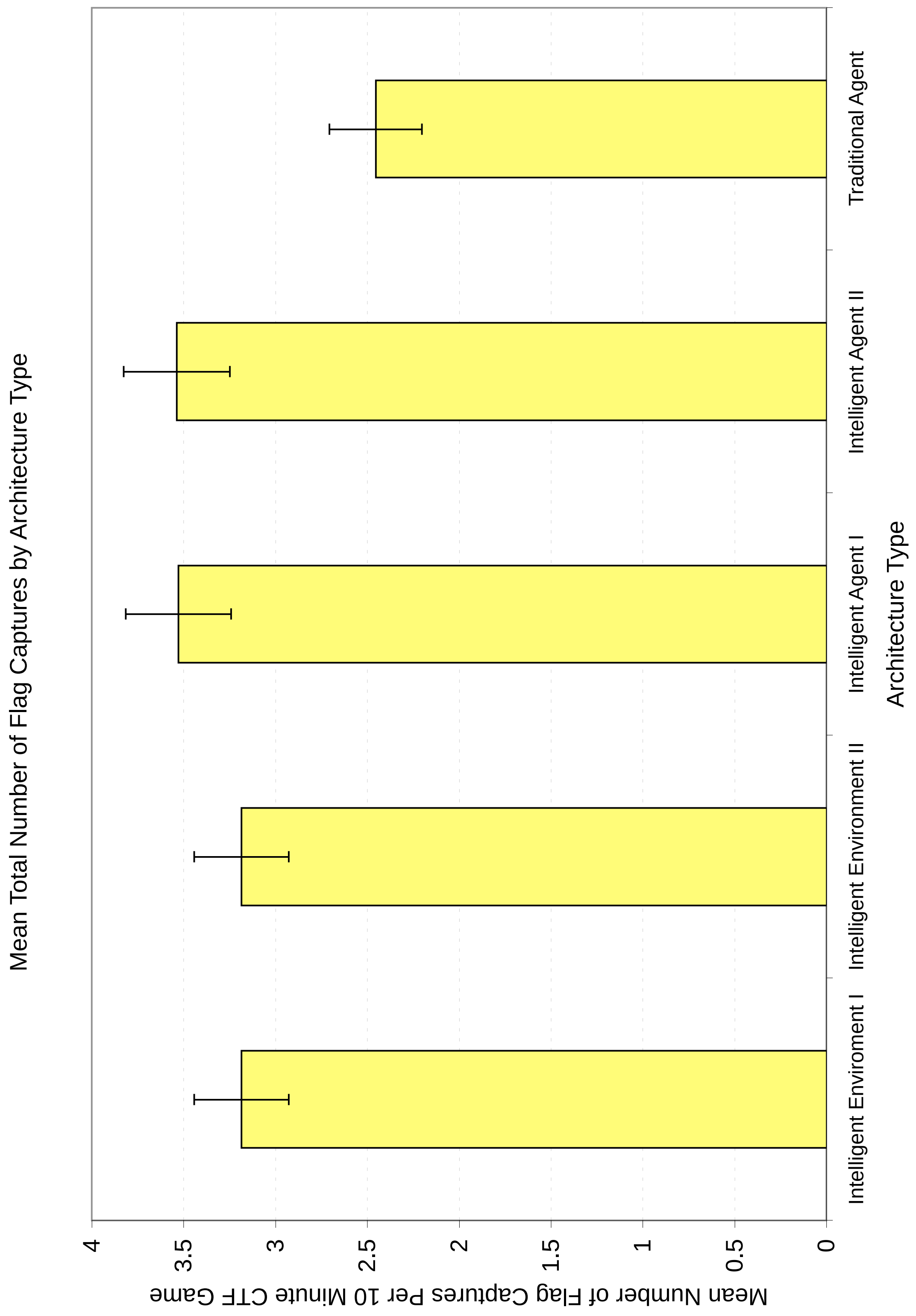
6.9.2 Architectural Complexity

In order to compare the computational run time complexity of the different architectures, all the games were run using a deterministic profiler. In this particular case since the

[62]A flag capture is a major event in a game of CTF that is only likely to occur a few times during a 10 minute game.



Figures 6.10: Mean Number of Flag Captures by Side and Architecture Type.



Figures 6.11: Mean Number of Flag Captures by Architecture Type

multi-agent simulation was written using the Python programming language, the profiler used was Hotshot, a high performance deterministic profiler that is part of the Python standard library.

The profiler logs various statistics about the execution of the simulation, providing information about which functions/methods were called, how many times they were called and the total computation time.[63] The profiler logs distinguish between the total time spent in a given function (excluding time spent in calls to sub-functions) and the cumulative computation time of a function which includes the time spent in a particular function and all sub-functions. A total of 540 profiler log files were generated. These were post-processed and the results plotted in a number of charts, separated by architecture type.

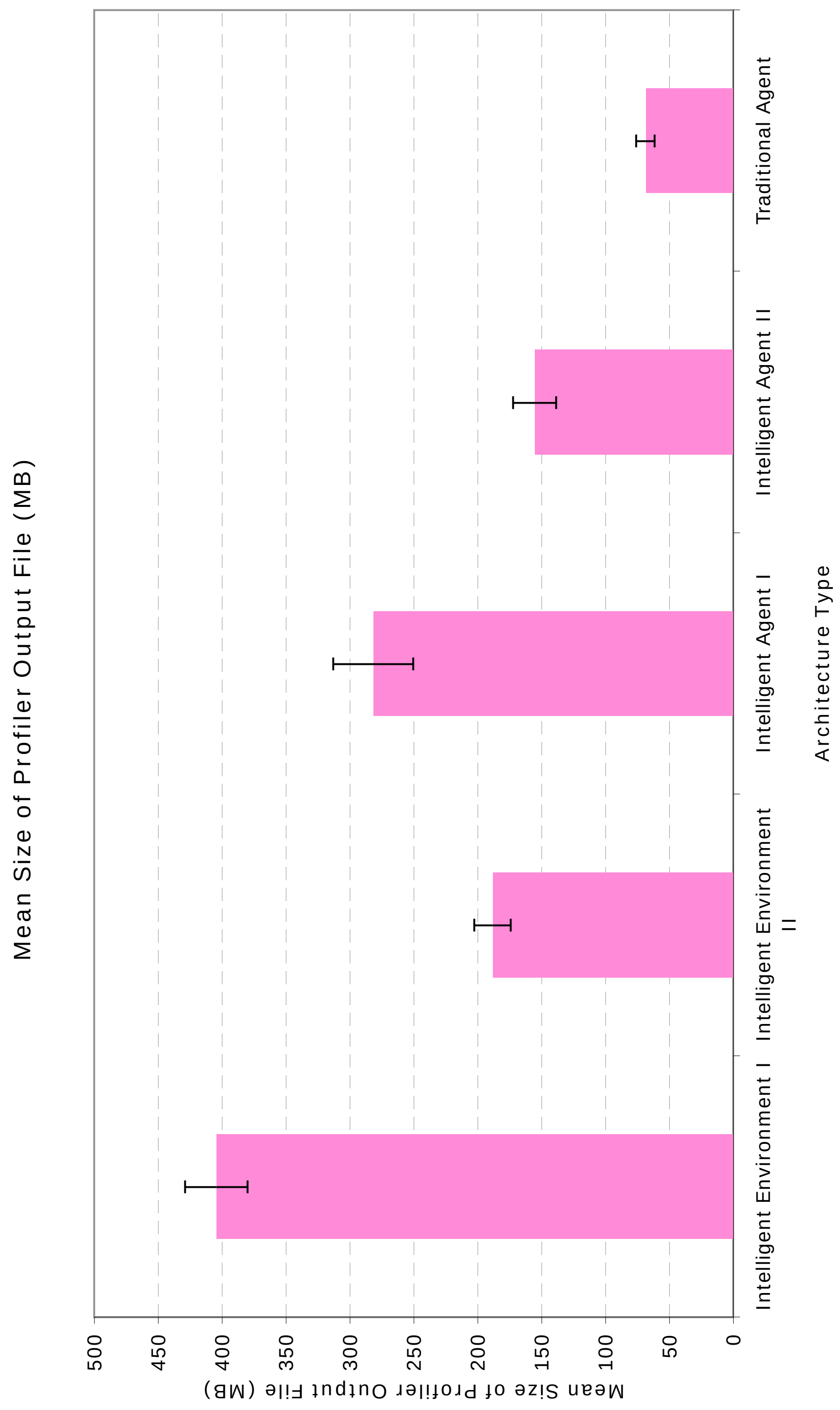
Each profiler log file generated by the profiler contained information about every single function call made within each ten minute capture the flag game simulation. An increase in the number of function calls corresponds to an increase in the size of the log file. Therefore, the size of the profiler's log file provides an indication as to the computational complexity of each architecture type.

Chart 4 (Figure 6.12)**. *Mean Profiler Log File Size*** *This chart shows the mean size of the profiler log file (in MB) for each architecture type. As in previous charts, the error bars indicate the standard deviation from the mean. This chart was generated by taking the mean of the profiler log files for each architecture type. Each bar in the chart represents the mean value from 108 CTF simulations.*

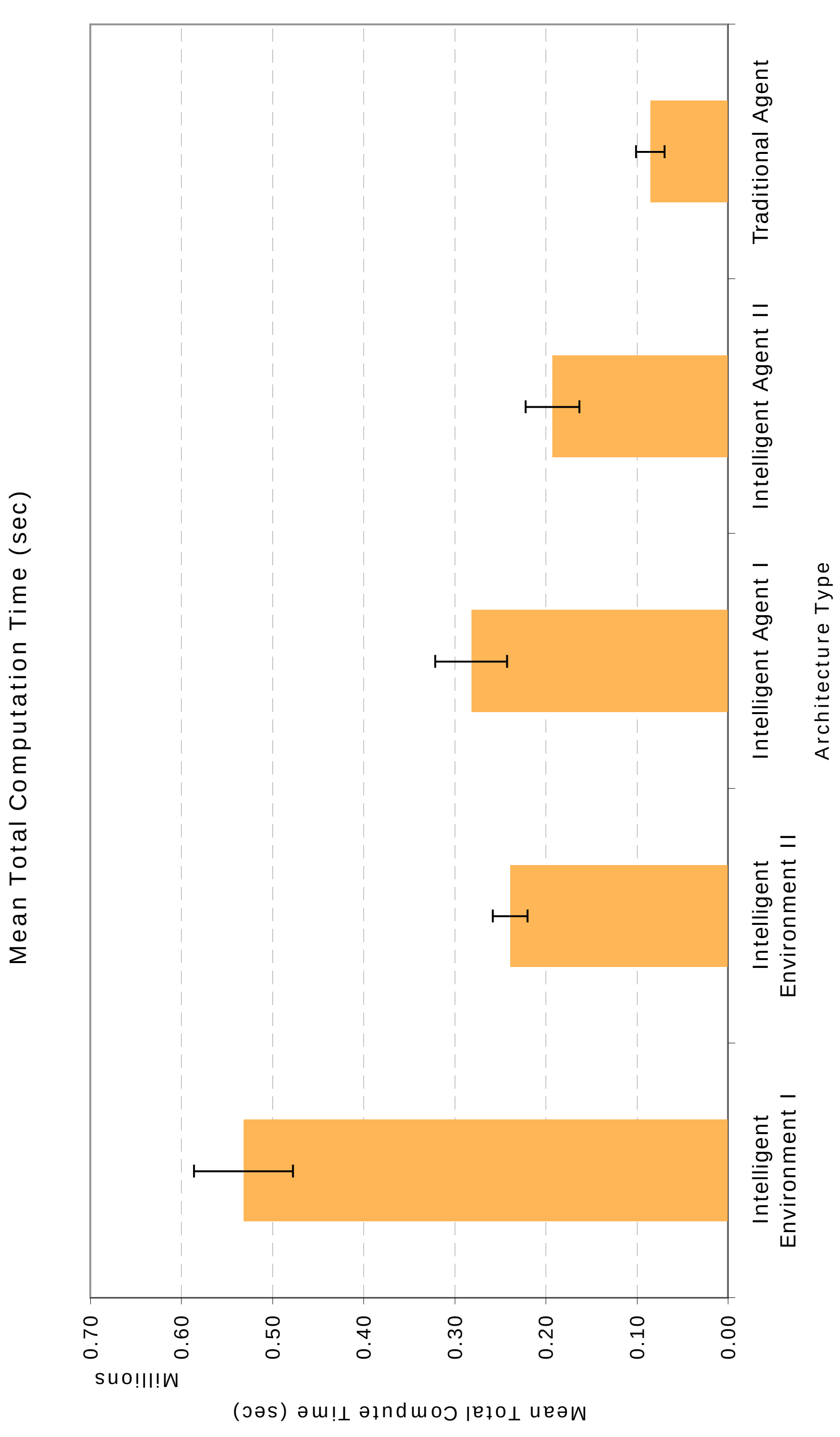
Chart 5 (Figure 6.13)**. *Mean Total Computation Time*** *This chart shows the mean total computation time (in seconds) for each architecture type, with the error bars indicating the standard deviation from the mean. Each column represents the average computation time of 108, 10 minute CTF games.*

Chart 6 (Figure 6.14)**. *Mean Cumulative Computation Time*** *This chart shows the mean cumulative computation (in seconds) for each architecture type, with the error bars indicating the standard deviation from the mean. Each column represents the average cumulative computation time of 108, 10 minute CTF games.*

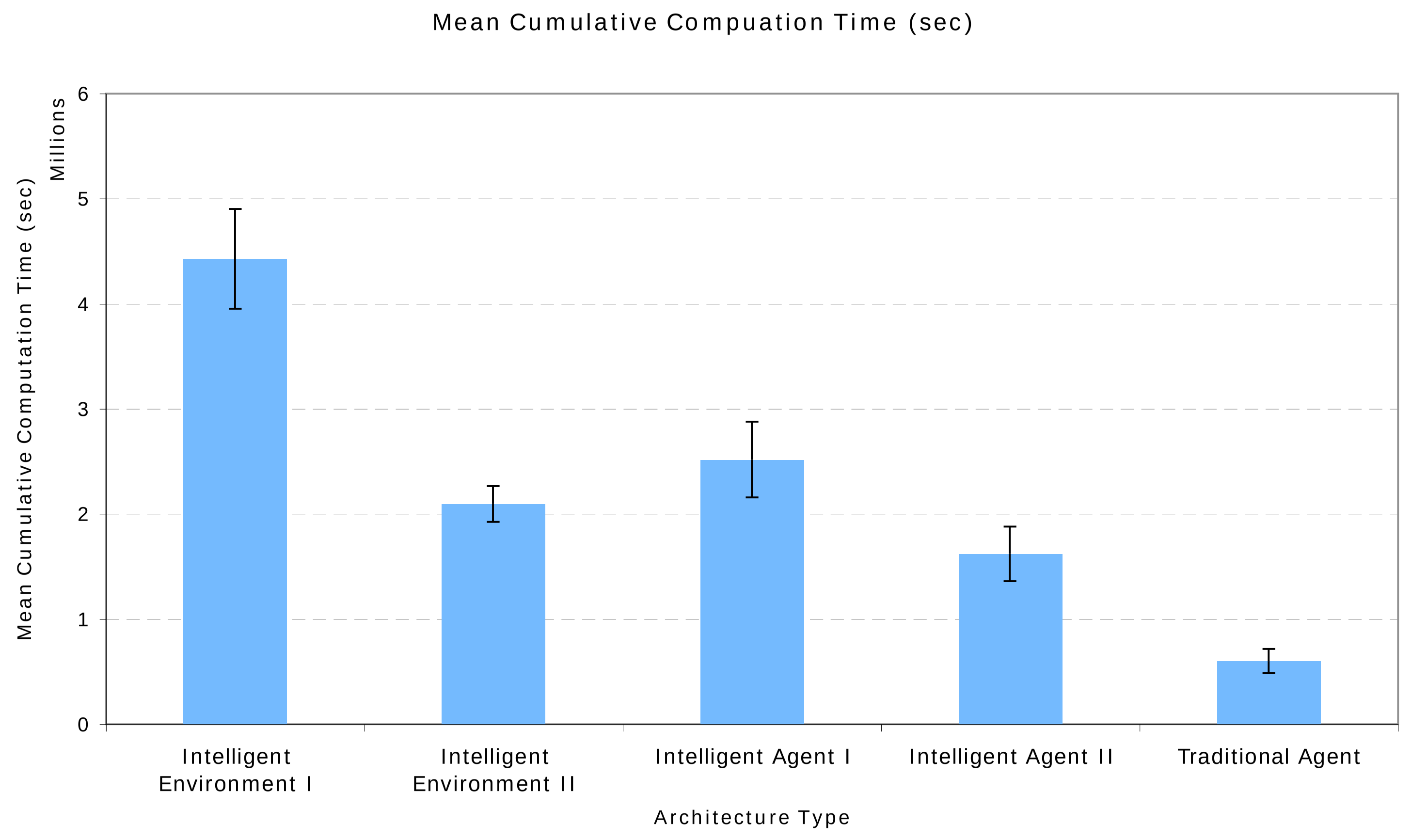
[63]The Python Hotshot profiler records functions and object method calls in the same manner. From a logging perspective they are considered equivalent, even though in the CTF the overwhelming majority of calls are method calls on objects.



Figures 6.12: Mean Size of Profiler Log File (MB)



Figures 6.13: Mean Total Computation Time (sec)



Figures 6.14: Mean Cumulative Computation Time (sec)

Chart 7 (Figure 6.15)**. *Mean Number of Total Function Calls*** *This chart shows the mean number of total function/method calls made for each architecture type, with the error bars indicating the standard deviation from the mean.*

There are a number of observations that can be made from the results shown in Figures 6.12, 6.13, 6.14 and 6.15.

Observation 1. *There is a relative similarity between architecture when comparing across the four metric (profiler log size, total computation time, cumulative computation time and number of function/method calls.*

The first observation is the relative similarity between the different architecture types across the four charts. Despite different metrics being charted (profiler log size, total computation time, cumulative computation time and number of function calls) it is to be expected that the relationship between the metrics are similar across architecture types because of the strong correlation between these metrics.

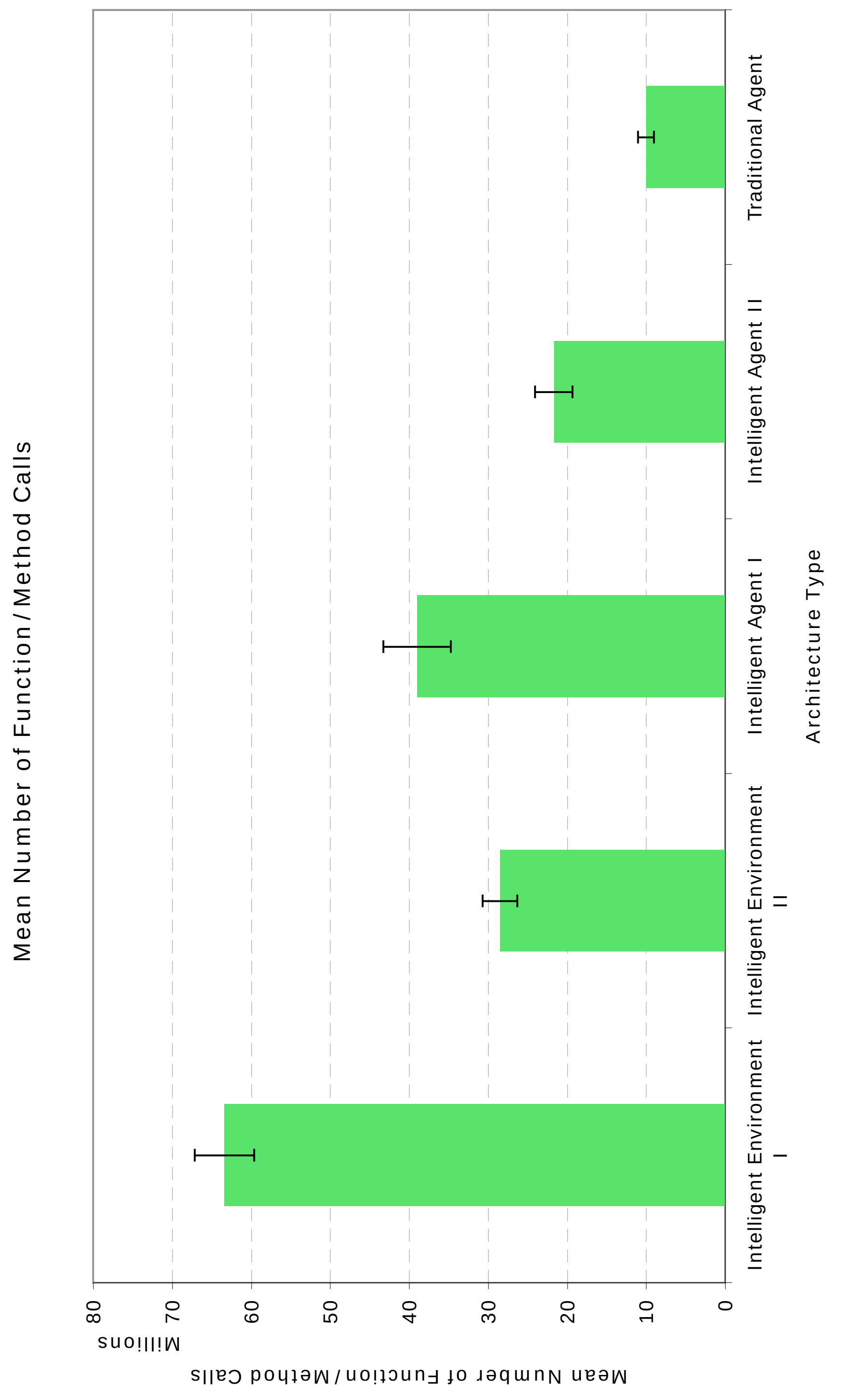
Observation 2. *The Traditional Agent architecture exhibits the smallest computational footprint across all four metrics being considered.*

The second observation is the relatively small values obtained across all four metrics for the Traditional Agent architecture when compared with the four affordance based architectures. This indicates that in the case of these CTF multi-agent simulations, the Traditional Agent architecture exhibits by far the smallest computational footprint in terms of computation time and number of function calls.

This is an expected result because, although this agent can adequately play a game of CTF, it exhibits simpler characteristics in both design and behaviour when compared to the affordance based architectures. More importantly however the Traditional Agent architecture lacks the richer and more complex interaction between the agent and the environment that is present in the affordance based architectures.

Observation 3. *Difference in computational footprint between Intelligent Environment and Intelligent Agent architecture types.*

The third observation is the difference in computational footprint between the Intelligent Environment and Intelligent Agent architectures. This difference is particularly pronounced when a comparison is made between architectures Intelligent Environment and Intelligent Agent across all the metrics presented. This is despite similar game outcomes (see Figure 6.11) and an identical behavioural repertoire for the tank driver agents across all



Figures 6.15: Mean Number of Function/Method Calls

affordance based architectures. The significant difference between computation time and the number of function calls between the Intelligent Environment and Intelligent Agent architectures is a consequence of how affordances are considered in both the environment and the agent.

In the architectures where affordances are computed in the environment, affordances are considered for each agent (in this case tanks) with respect to every single other entity and agent in the environment. As the number agents, entities and affordances to consider increases so does the computational overhead. In comparison to the Intelligent Agent architecture the agent only evaluates affordances with respect to other entities in the environment it has knowledge (or a belief) about.

This is largely restricted to the other entities that the agent can perceive using its sensors and to any other entities which the tank driver agent has beliefs about, such as the location of enemy and friendly flag bases. As one can imagine, since the agent only has to evaluate affordances with respect to a subset (primarily perceivable) entities in the world, as compared to every single other entity in the world, the number of affordance evaluations is significantly reduced and so is the computational footprint of this architecture.

This observation provides an important example of the trade-off between closely adhering to the concept of affordance as espoused by ecological psychology and the practical engineering concerns such as computational performance required in multi-agent simulations used in real world application domains. In this example the Intelligent Environment architecture was closer in concept to the psychological view of an affordance. That is, an affordance is not an internal agent mental attitude, agents can directly perceive affordances in the environment and affordances can exist with respect to entities an agent cannot directly perceive.

Consequently there was a higher computational overhead when compared to the Intelligent Agent architecture which treated an affordance as an internal agent mental construct and only evaluated affordances with respect to entities it could perceive with its sensors. Which approach a designer of a multi-agent simulation chooses will depend on the exact requirements of the domain being modelled.

Observation 4. *There is a reduction in computational footprint (faster computation) times when comparing the general versions of the architectures (Intelligent Environment I (General)and Intelligent Agent I (General)) with the more specific versions (Intelligent Environment II (Specific) and Intelligent Agent II (Specific)). The architectures which make use of the more specific version of the affordance algorithm have a lower overall computation time.*

The fourth observation that can be made from these figures is the reduction in computational overhead in going from Intelligent Environment I (General) to Intelligent Environment II (Specific) and in going from Intelligent Agent I (General) to Intelligent Agent II (Specific) architectures. In both cases domain specific information was used to make some intelligent decisions about which affordances to evaluate with respect to which entities. The original versions of the affordance architectures Intelligent Environment I (General) and Intelligent

Agent I (General) can be considered as general purpose affordance architectures that evaluated all affordances with respect to all entities in the world equally.

While a generality has a number of advantages such as elegance and wider applicability it is possible to improve computational performance by taking into account some domain specific information. This was the approach taken in architectures Intelligent Environment II (Specific) and Intelligent Agent II (Specific). In these cases the list of available affordances were partitioned into a number of sets (those that were applicable to tanks, those that were applicable to flags and those that were applicable to bases). This meant when the affordance generation algorithm was considering the affordances with respect to a particular entity, the type of the entity (tank, flag or base) was used to determine which set of available affordances would be evaluated.

Therefore, the evaluation of whether a flag afforded being attacked by a tank was never made because it was not relevant. Additional domain specific information was also taken into account such as the relative side (blue or red) of specific entities. This domain specific information allowed for the number of affordance evaluations to be significantly reduced and hence the total computation time and number of function calls to be reduced.

What is also important to note is that these reductions in affordance evaluations due to domain specific information could be done without having any impact on the outcome of the games. Figures 6.9, 6.10 and 6.11 show that the number of tank kills and flag captures is the same when comparing the two Intelligent Environment architectures and the two Intelligent Agent architectures. Furthermore, an analysis of the individual kills in each of the 108 runs show an identical outcome for the general purpose and the optimised architectures.

This observation also provides an example into the trade-off between a general purpose affordance based architecture that can be used across many different types of multi-agent simulations as opposed to an affordance based agent architecture which has used domain specific information to optimise computational performance.

6.9.3 Architectural Subsystems

The information obtained from the profiler logs was further processed to provide additional information about the relative computation in the multi-agent simulation organised by sub-system. Every single Python module [64] in the CTF simulation was partitioned into one of five major architectural sub-systems. Each sub-system represented major functionality required by the CTF multi-agent simulation. The five major sub-systems were Simulation, Entity, Agent, Environment and Utilities as described in Section 6.4.

For each function/method, in addition to data such total computation time and number of calls, the profiler logs also recorded in which module the function belonged. This made it relatively straightforward to organise the results from the profiler by sub-system. This not only allowed for the mean number of function calls and mean total computation time to be charted for each architecture type as was shown in Section 6.9.2, but to also break down these metrics according to sub-system. This meant that for each architecture type the relative computation time spent in each sub-system could be easily compared.

[64] A Python module is a collection of related functions and classes.

Chart 8 (Figure 6.16). ***Mean Number of Function Calls by Subsystem (Stacked Bar)*** *Figure 6.16 charts the mean number of function calls for each architecture type using a stacked bar chart. For each architecture type the stacks in each bar shows a breakdown of the mean number of function calls attributed to each of the five subsystems being considered.*

Chart 9 (Figure 6.17). ***Mean Number of Function Calls by Subsystem (Relative Stacked Bar)*** *Figure 6.17 displays the same data except it is presented in a relative stack bar chart. This indicates the mean number of function calls for each subsystem in percentage terms relative to other subsystems, for each architecture type.*

Chart 10 (Figure 6.18). ***Mean Computation Time by Subsystem (Stacked Bar)*** *This figure charts the amount of mean total computation time spent in each subsystem in each architecture type.*

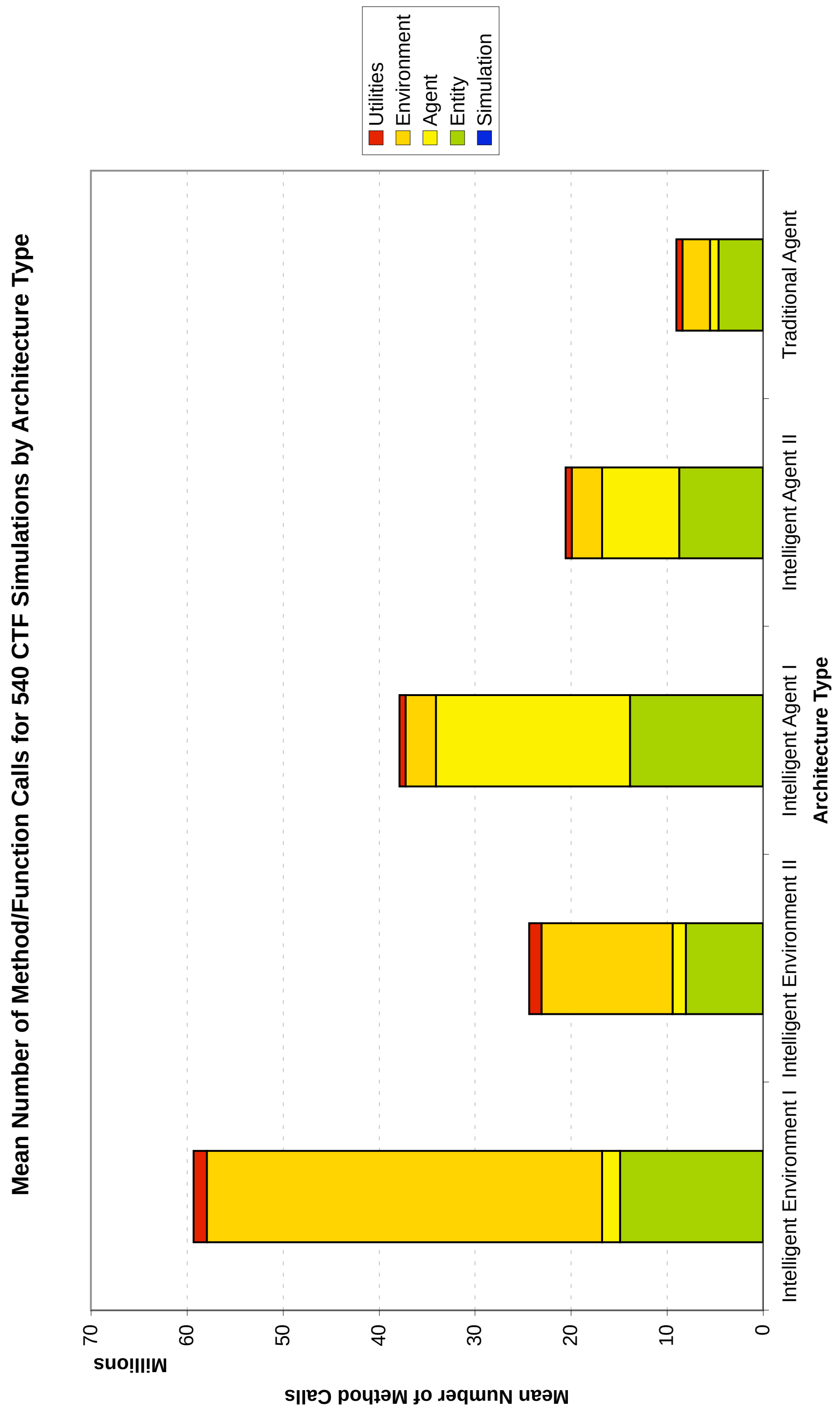
Chart 11 (Figure 6.19). ***Relative Mean Computation Time by Subsystem (Relative Bar)*** *This figure charts the mean total computation time as a relative proportion of computation time spent in each subsystem, in each architecture type.*

Figures 6.18 and 6.19 chart the mean total computation time for each architecture type, split up by subsystem in the same manner as Figures 6.16 and 6.17. These charts provide a clear indication as to how much computation time was spent in each subsystem for each architecture type for both absolute and relative perspectives.

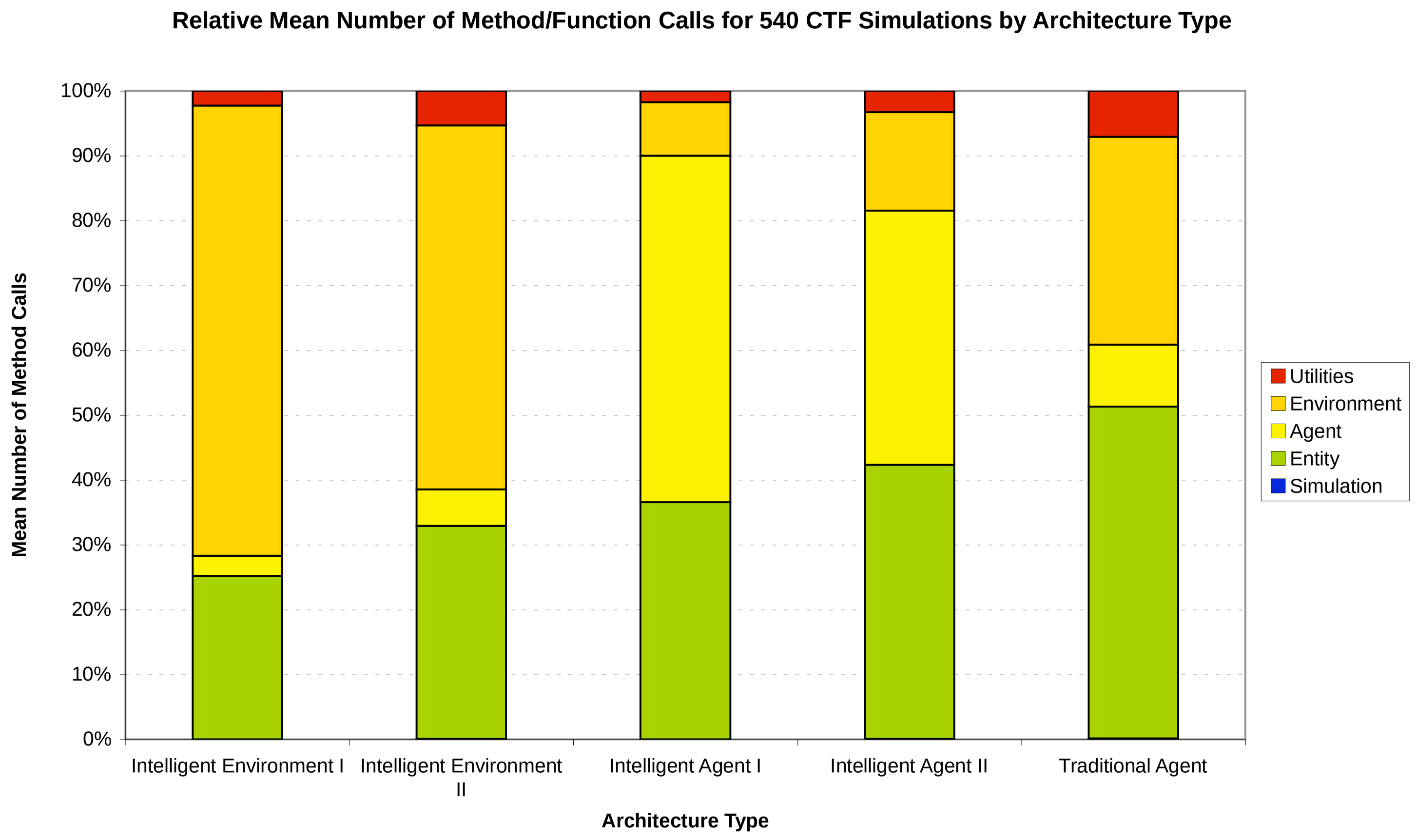
A number of observations can be made about these results.

Observation 5. *Contribution of the Simulation subsystem to the total computation time and to the total function/method count is negligible for all architecture types.*

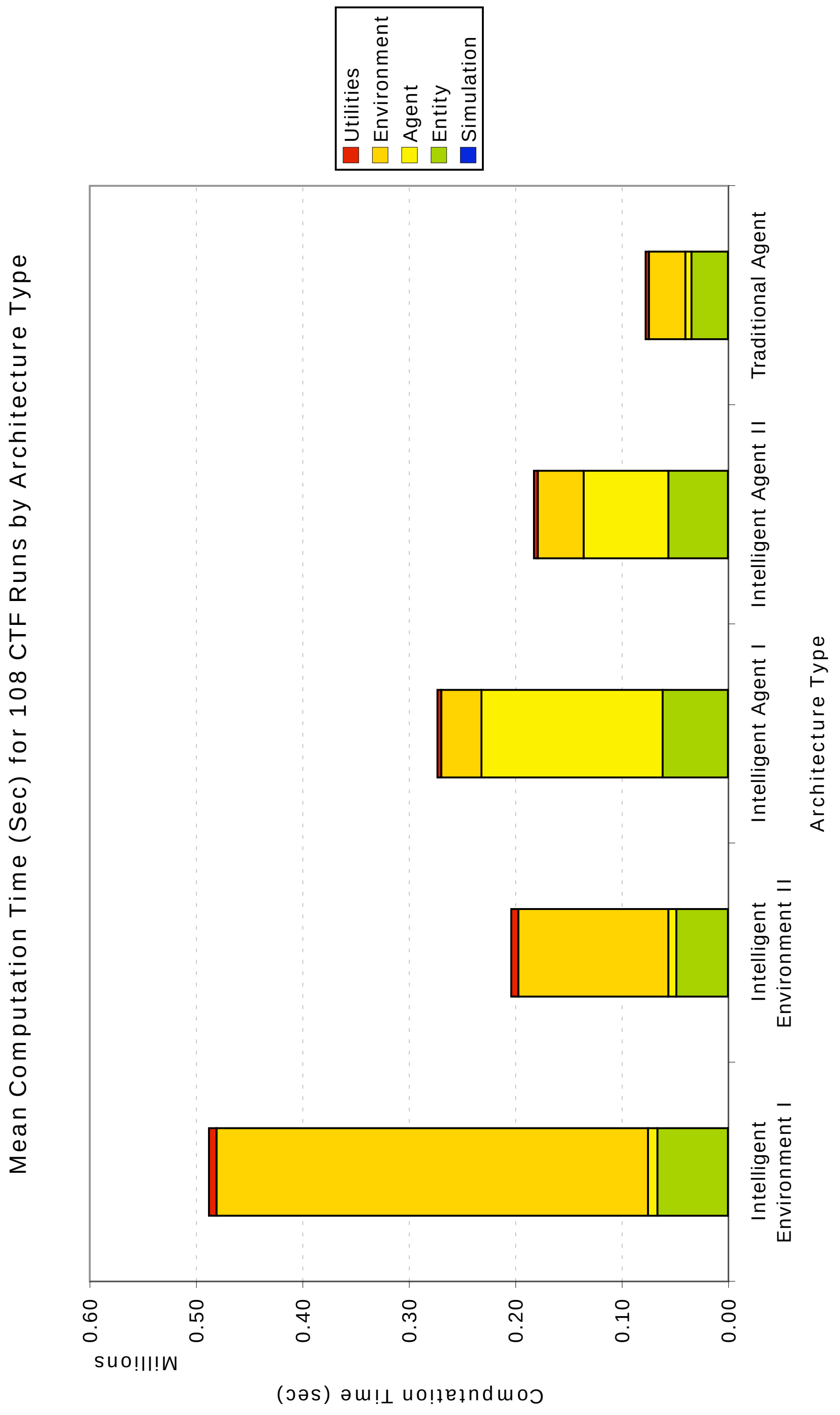
It appears that the Simulation subsystem is not represented in the charts. Although both the number of function calls and the computation time associated with the Simulation subsystem is finite, it is sufficiently small relative to the other subsystems to not register a sufficiently large proportion of the bar charts shown. This is to be expected because the Simulation subsystem is quite small and its functionality is limited to executing the entity and environmental models in a time-stepped fashion.



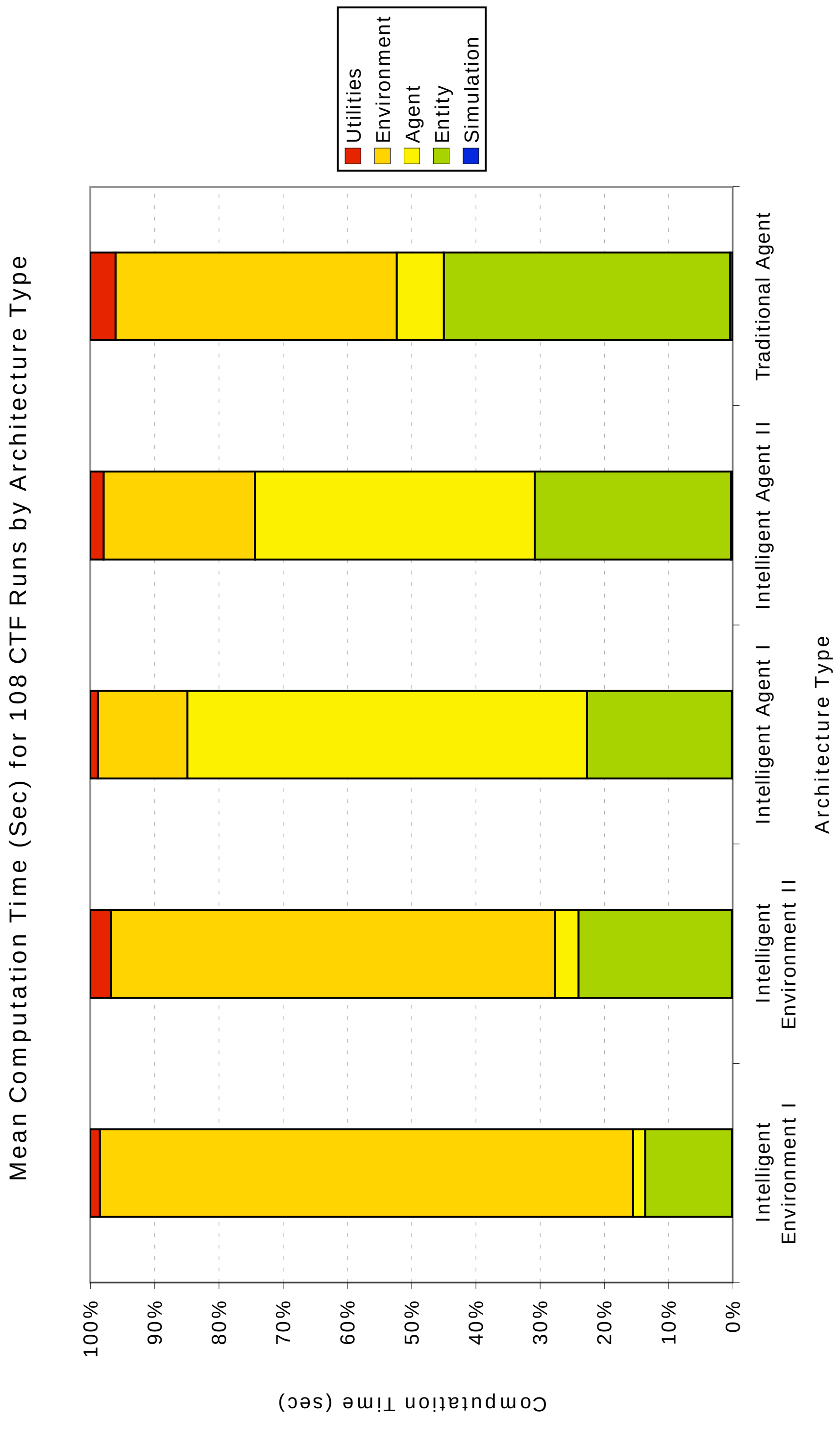
Figures 6.16: Mean Number of Function/Method Calls by Architecture Type.



Figures 6.17: Relative Number of Function/Method Calls by Architecture Type



Figures 6.18: Mean Computation Time by Architecture Type.



Figures 6.19: Relative Computation Time by Architecture Type

Observation 6. *Contribution of the Utilities subsystem to the total computation time and to the total function/method count is relatively small across all architecture types.*

The next observation is the relative small amount of computational effort spent in the Utilities subsystem across all architectures. Again this was to be expected as the calls to general purpose utility functions were independent of any particular architecture types.

Observation 7. *In the Traditional Agent architecture, the computation time and proportion of total function/method calls is dominated by the Entity and Environment subsystems.*

The next observation one can make is about the computational footprint of the Traditional Agent architecture. The largest computational effort consists of code in the Entity (vehicle, sensor, weapon) subsystems followed by the Environment subsystem. The simple behavioural complexity of the tank driver agent is reflected in the small amount of computation devoted to the Agent subsystem.

Observation 8. *In the Intelligent Environment I (General) and II (Specific) architectures it is the Environment subsystem which is the largest contributor to the total computation time and the total number of function/method calls.*

Observation 9. *In the Intelligent Agent I (General) and II (Specific) architectures it is the Agent subsystem which is the largest contributor to the total computation time and the total number of function/method calls.*

The next and perhaps most important observation is the dominant subsystem in terms of computational effort in the four affordance based architectures. In the two Intelligent Environment architectures it is the Environment subsystem which dominates, while in the two Intelligent Agent subsystems it is the Agent subsystem which dominates computationally. This is to be expected due to the design of architectures and is the reason why the affordance based architectures were named in this manner.

In both the Intelligent Environment architectures the Environment subsystem dominates the computation. This is because all the decisions about what action possibilities or affordances are available to all agents are computed by the environment. The environment in this case is *intelligent* because it is computing the courses of action available to it.

Traditionally this type of computation would be undertaken within an intelligent agent, and would typically correspond to the process known as epistemic reasoning; that is reasoning about the world. The tank driver agent in this case is comparatively simple, in that it is presented its available options (or more precisely affordances) from the environment. The agent's job is then to reason or to decide about which affordance to adopt next. This relatively smaller computational effort is reflected in the smaller computational representation in all the charts associated with the Agent subsystem in the two Intelligent Environment architectures.

On the other hand the opposite is true (as expected) for the two Intelligent Agent architectures. In this case it is the agent's responsibility to conduct all computation associated with epistemic and procedural reasoning. In other words, not only does each tank driver agent have to determine which affordances are available to it but it also must decide which affordance to adopt.

Although this is an affordance based architecture, all information regarding reasoning about affordances are undertaken within the agent. Consequently the Agent subsystem dominates the computation in the Intelligent Agent architectures. The name for these architectures is again applicable because all the computation traditionally associated with *intelligence*, such as determining what actions are available and then deciding which one to adopt, is adopted by the agent.

It is important to note that the Intelligent Environment and the Intelligent Agent architectures represent the extreme cases. In one case all the computation associated with determining what affordances are available are undertaken in the environment and in the other case they are undertaken within the agent. As mentioned earlier, there is a trade-off between close adherence to a model of affordance more closely aligned to that in ecological psychology, and the practical engineering considerations arising from developing a multi-agent simulation for a real world application domain.

Observation 10. *In both the Intelligent Environment and the Intelligent Agent architectures there is a significant improvement in computation time and a reduction in the total number of function/method calls when comparing the architectures implementing the more specific version of the affordance algorithm (Intelligent Environment II (Specific) and Intelligent Agent II (Specific)) when compared to the more general purpose implementation (Intelligent Environment I (General) and Intelligent Agent I (General))*

The next observation relates to the improvements in computation time made in the affordance based architectures. As described earlier, the reduction in computation time in going from Intelligent Environment I (General) to Intelligent Environment II (Specific) and from Intelligent Agent I (General) to Intelligent Agent II (Specific) were made possible by considering domain specific information to reduce the number of affordance evaluations.

In the case of the Intelligent Environment architectures these improvements were made in the Environment subsystem. Given the significant amount of computation taken up by the Environment subsystem, by reducing the number of affordance evaluations reduced the amount of time spent in this subsystem and hence significantly reduced the total computation time. Similarly in the case of the Intelligent Agent architectures, the reduction in the amount of computation time spent in the Agent subsystem allowed the total computation time for the Intelligent Agent II (Specific) architecture to be significantly reduced.

6.10 Discussion

This Chapter explored the impact of different affordance based agent-environment interaction architectures in the context of a multi-agent simulation implementing a game of Capture The Flag. Specifically, two extreme cases were looked at.

The first case involved the computation of affordances being undertaken in the virtual environment. This architectural variant was labelled as the Intelligent Environment because the computation involved determining what actions (or affordances) were possible in the world and were undertaken in the environment for each agent. This off-loaded some of the computational processing from the agent resulting in a much less complex agent design.

The second case involved the computation of affordances being undertaken individually by each agent. This meant that each agent was more sophisticated in both design and the level of computational intelligence which it was required to capture, and hence was labelled as an Intelligent Agent architecture.

By comparing these different affordance based architectures it is possible to draw some conclusions and learn some lessons about designing such multi-agent simulations. However, it is also important to note that the lessons learnt and the conclusions drawn from the development and experiments undertaken using this CTF multi-agent simulation cannot necessarily be generalised and be applied to all multi-agent simulations or even all multi-agent simulations making use of an affordance based approach to agent-environment interaction.

While some of the conclusions drawn can be applied to other multi-agent simulations and application domains, each multi-agent simulation developed will be guided by its own unique system requirements and the nature of the specific application domain.

One of the important lessons to come out of an affordance based implementation of a CTF multi-agent simulation is to show the viability of an affordance based approach. The multiple affordance based architectures, in conjunction with the affordance based approach used in the HAVE simulation described in Chapter 2 shows the potential for using affordances as a mechanism for modelling agent-environment interaction in multi-agent simulations.

One of the advantages of an Intelligent Agent architecture is that the concept of an affordance can be introduced into a legacy multi-agent simulation in a much easier manner. This is because the introduction of affordances is isolated in each individual agent and there is less need to alter the aspects of the simulation representing the virtual environment. This means that there is less coupling between different architectural subsystems and hence the complexity of the architectural design is not significantly impacted.

However, the most significant disadvantage is the increase in the complexity of the design of the agent. This is because each agent has to determine for itself which affordances are available to it instead of directly perceiving them in the environment, in addition to deciding which affordances to adopt. There is also a question of agent scalability both from the perspective of an agent designer and an agent programmer. As the complexity of a multi-agent simulation increases [65] the potential for the complexity of the agent design and corresponding code base to increase as the requirement for additional agent behavioural complexity increases.

An agent centric view of affordance also means that the agent can only determine affordances with respect to entities that the agent can perceive with its sensors. Additionally, an agent may possibly consider affordances with respect to entities that it has beliefs about even if it cannot directly perceive them.[66] This is in contrast to the view of affordance as a directly perceivable action possibility relating agent and environment, where affordances can exist between an agent and an entity in the world – independent of whether the agent can perceive or is aware of the affordance.

Modelling affordances using an Intelligent Environment based approach has a number of advantages. The first is that there is no one architectural sub-system responsible for the agent's computational behavioural intelligence. The responsibility for intelligent behaviour is located in both the environment and in the agent subsystems, and more importantly arises from the interaction between the two.

From a practical software engineering perspective however, it also means that the design of the agent can be kept relatively simple. That is the agent does not have to compute what action possibilities are available to it. Rather, it simply has to determine which if any affordances to adopt and then carry out the appropriate actions in the world. This type of reasoning is well suited to plan based procedural agent reasoning languages such as the BDI family of agent programming languages.

A design where affordances are computed in the environment (or more precisely in the interaction between agent and environment) is much more natural because the different types of computation which contribute to intelligent behaviour are located in the subsystems which are appropriate. This results in a cleaner design with responsibility for intelligent behaviour allocated to the parts of a multi-agent simulation architecture in which it is most suitable. In comparison an architecture in which all responsibility for intelligent behaviour is allocated to an intelligent agent runs the risk of downplaying the important role in which environments play in many multi-agent simulations.

This type of architecture can be considered a type of annotated virtual environment, in which the annotations are not static labels. Rather the annotations are dynamic, computed by the environment, tailored for each agent, and most importantly they are action oriented. That is, from one perspective a virtual environment which determines affordances for the agents that are situated in it can be considered an annotated intelligent environment. The difference being that the annotations are affordances that have been generated by the environment at simulation run time.

An Intelligent Environment approach to modelling affordances also implies that the design of

[65]There are many ways in which the complexity of a multi-agent simulation can increase. The most obvious ways are by increasing the number of entities, the number of agents and by increasing the complexity of the virtual environment in which the agents and entities are situated, which can mean increasing the complexity of the relationships between agents and other entities.

[66]For example an agent may perceive an affordance with respect to an entity that is temporarily hidden from view.

the virtual environment becomes more complex. In the CTF simulation this was handled by making the design of the environment multi-modal. This has a number of advantages.

The first is that the design of the virtual environment is modular, and environmental modalities can be added and removed as needed. The second is that a multi-modal environment captures the complex aspects of the real environment, which goes beyond the physical aspects and includes abstract aspects such as social and organisational environments. This is important because the affordances humans perceive are dependent on many different environmental modalities.

In the case of the CTF simulation, the computation of affordances was considered as another environmental modality. This modality was a view of the world from the perspective of the affordances available to the different agents in the game. This *affordance space* could be added to a game scenario when an Intelligent Environment architecture was specified and removed when an Intelligent Agent architecture was being used.

More importantly however, although the affordance modality is just one of the aspects of the virtual environment which make up the total CTF world, it is also a component which acts as interface between the agent and the rest of the virtual environment (e.g. the other modalities). From this perspective this architecture reflects the affordances as a relational concept between agent and environment.

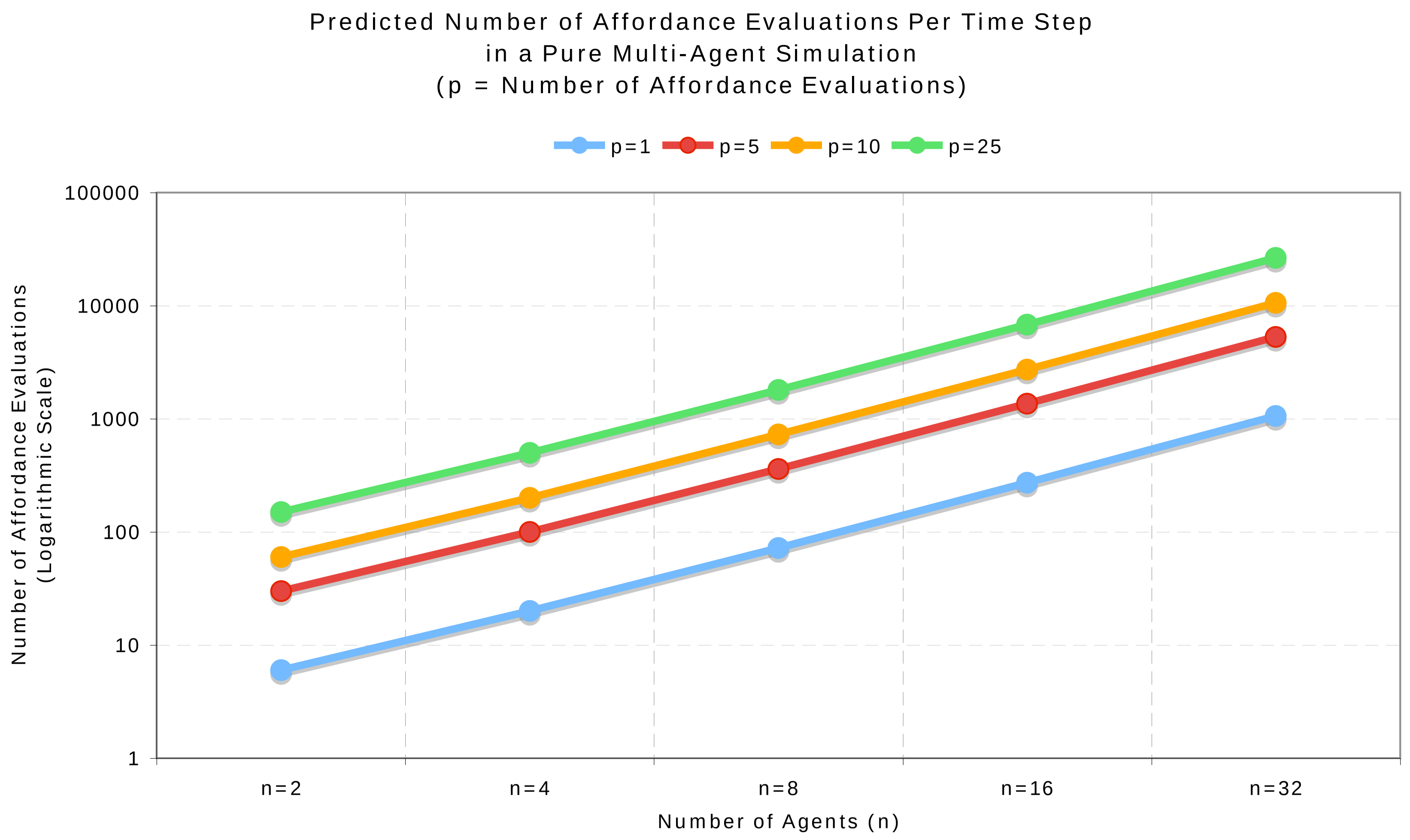
6.11 Future Experimental Work

The development of the CTF simulation and the corresponding affordance oriented interaction architectures allows for a qualitative comparison of the impact an affordance based approach has on the design of these simulations. A discussion on the design impact which incorporates the lessons learnt from the development of the HAVE simulation is presented in Chapter 7.

From the perspective of the experimental work described there is the prospect of conducting further experiments to explore a number of different avenues.

One avenue of investigation is to look at increasing the number of entities in the simulation to see how the computational performance of the simulation is affected. The number of agents and entities in the experiment were deliberately kept small. A game of 2 v 2 was selected because it represented the simplest possible form in which a reasonable game of CTf could be played. This underlies the entire purpose of selecting the CTF domain as it represented a cut down domain that could be used to analyse affordance oriented agent – environment interactions without the overhead of a more complex application domain such as the Close Air Support domain described in Chapter 2.

The number of entities can be increased in a number of different ways. The most obvious way is to increase the number of agents in the game. One can consider games with four, six, eight (or even greater) number of entities on each side. One can also consider scenarios with an asymmetric number of tanks. The architecture does not change as the number of agents are increased in the simulation. However the impact on the computation time can be predicted. The greater the number of agents (and hence entities) in the simulation the greater the number of affordance evaluations required. The algorithmic analysis described



Figures 6.20: Predicted number of affordance evaluations in a pure multi-agent system (where all entities are agents) as the number of agents n increase and the number of affordances p also increase. Note the vertical axis is using a logarithmic scale.

in Section 5.9 in Chapter 5 can provide an indication as to how increasing the number of agents increases the number of affordance evaluations. Some example numbers are provided in Table 5.1.

Figure 6.20 shows how the number of affordance evaluations can increase as the number of agents n in the simulation increase and as the number of affordances p being considered increases. The graph shows a logarithmic scale. While this figure illustrates the case for a pure multi-agent system it is obvious that as the number of agents increase the number of evaluations increases significantly. This will have a significant impact on the computational performance of the simulation.

However, there are two important notes to make. This is the theoretical worst case scenario. What hasn't been taken into account is the reduction in the number of evaluations which can be made by taking domain specific information into account in the affordance generation algorithms in an multi-agent simulation as was done in the Intelligent Environment II (Specific) and Intelligent Agent II (Specific) architecture types.

The second point relates to the application domain in which these type of affordance architectures are to be used. They are suitable for domains where there are a relatively small number of heavyweight intelligent agents. This is certainly the case in simulations of air combat where the number of aircraft (and hence fighter pilot agents) are typically in the range of two, four, eight and in some cases up to sixteen.

This leads to another avenue of exploration, namely that of optimisation of the number of affordance evaluations being undertaken. By exploring alternative techniques for reducing the number of evaluations these type of affordance architectures may become viable for larger number of agents.

Further experimentation can be performed by increasing the complexity of the environment. The complexity of the environment can take on a number of forms. The most obvious one is to introduce other non-agent entities in the environment in which either hinder or assist the tank agents. These can include walls, doors and other obstacles. The other form of complexity which can be studied is that of social complexity. By introducing more complex social command and control structures in large teams it becomes possible to evaluate affordance evaluation in the context of different command and control relationships.

Finally, additional scope for experimentation exists in the definition of the affordances used in the simulation. This can include exploring not only the number of affordances each agent considers, but also their complexity.

6.12 Summary

This Chapter presented an evaluation of the model of affordance presented in Chapter 5. This was done by designing and implementing the affordance based model of interaction in a multi-agent simulation of the game Capture The Flag.

Two major design variations were looked at. In the first, affordances were computed in the environment, while in the second affordances were computed in the agent. This allowed a number of experiments to be conducted to evaluate the architectural designs in a number of areas such as the impact on computational performance.

In Chapter 7 the lessons learnt and insights gained from the development of the affordance model in Chapter 5 and the two affordance based multi-agent simulations, (HAVE described in Chapter 2 and Capture The Flag described in this Chapter) are brought together and discussed in the context of developing affordance based multi-agent simulations.

Chapter 7

Discussion

"The environment is everything that isn't me."

— Albert Einstein

7.1 Introduction

This Chapter discusses the issues associated with modelling affordances in multi-agent simulations in the context of lessons learnt and insights gained from the development of the affordance model described in Chapter 5 as well as the two implementations, Capture The Flag (CTF) and The Human Agent Virtual Environment (HAVE), in Chapters 6 and 2 respectively.

The discussion starts by looking at some of the issues associated with modelling affordances. such as the relation between the model of affordance as implemented in a multi-agent simulation and the concept of an affordance as described in ecological psychology.

This is followed by a discussion of the more practical issues associated with developing affordance based multi-agent simulations. In particular the impact that an affordance based approach has on a number of different areas is discussed. The areas looked at are the impact on design, the impact on computational performance and the impact on the application domain when these types of multi-agent simulations are deployed.

Alternative approaches and ways to extend the existing approach to modelling affordances are then briefly discussed.

7.2 On Simulating Affordances

Modelling and simulating the concept of affordances in a multi-agent simulation has presented a number of challenges. These challenges largely involved deciding:

1. What aspects of the theory of affordance to include in a computational model suited for multi-agent simulation.
2. How to incorporate ideas from affordance theory into existing models of agency.

3. How faithful to remain to the theory of affordance as described in ecological psychology, when faced with practical software engineering issues in the design of the multi-agent simulation.

Deciding on what aspects of the theory of affordance to consider for inclusion into a computational model of agency was influenced by a number of factors.

Perhaps foremost in consideration was the types of military multi-agent simulation which motivated this research. An affordance based model of agency needed to have the prospect of providing some practical utility and benefits to the developers and users of these types of simulation systems.

For example the real-time and faster than real-time [67] computational constraints of military multi-agent simulation meant that the model had to strike a balance between the appropriate level of fidelity and simplicity to allow for an acceptable level of computational performance.

Consequently concepts from ecological psychology that were related to affordances but were not suitable for types of multi-agent simulations being considered were not included in the model. The concept of visual or optical flow [54, 18] which deals with how humans and animals visually perceive their environment through motion was one such concept.

Another important consideration was to include the concepts from affordance theory which were relevant and useful to military multi-agent simulation. For example the relation between affordances and agent intentionality was an essential inclusion because of the important role in which intentions play in military decision making. It also meant that the model of affordance could be related and possibly integrated with existing models of agency such as the BDI model which has been used to model military decision making in a number of multi-agent simulations.

Similarly it was important to capture the relational aspects of affordances in the model, not only because it is a principal part of affordance theory but also because modelling relationships (such as command and control) are critical to understanding how decisions are made during a battle.

In deciding on how to incorporate the agent based model of affordance into existing models of agency a number of factors came into consideration. One such factor was deciding how affordance perception and consideration related to existing models of agent reasoning. The affordance model was framed in terms of two reasoning models – the BDI model of agent reasoning and Boyd's OODA loop model of military decision making.

Another factor was to consider how to incorporate the concept of affordances into the traditional *Perceive-Reason-Act* model of agent reasoning. Closely related to this was how to incorporate the relationship with the environment. The *Perceive-Reason-Act* model allows for an agent to be treated as separate and self-contained reasoning modules where percepts are fed in, a reasoning processes is undertaken and ultimately actions are sent to the environment.

However, the introduction of the affordance concept into the *Perceive-Reaso-Act* model meant that the interaction with the environment became more important. The relationship

[67]Faster than real-time simulation is critical for military operations research simulations which are often run as in batches of hundreds or thousands Monte-Carlo runs building up a statistical picture of mission outcomes in a given scenario.

with the environment became more complex and more situated. Interaction with the environment is not only something that happens at perception and action. Rather, the agent must reason about action possibilities – concepts which have been directly perceived in the environment.

Adopting an affordance based approach meant that the relationship between the agent and the environment needed to be reconsidered. The close alignment between perception and action and the provision for the direct perception of affordances in the environment had the potential to significantly impact on the design of a multi-agent simulation. This naturally raised the question of how faithful the computational model should be to the theory of affordances.

This was addressed by considering how the model was to be used in a multi-agent simulation. It was important to recognise that the aim was not to develop a full blown psychological model of affordance in order to explore and understand how humans perceive affordances. Taking such an approach would clearly be in the field of ecological psychology and would be beyond the domain and scope of this thesis. Rather, affordances were used as a mechanism for better understanding, explaining and hopefully improving the state of the art in the representation of agent – environment interaction in multi-agent simulations.

This stance allowed for the computational model to be developed in a way that allowed for flexibility in the resulting design and implementation. Although the computational model captured important aspects of the affordance concept (such as their relational nature, and the link to agent intentionality) it did not prescribe that affordances should only be directly perceived in the environment. This allowed for a number of different design implementations adopting different approaches.

In the Human Agent Virtual Environment described in Chapter 2, affordances were implemented within the concept of a traditional agent reasoning model, where the action possibilities each agent perceived were generated inside each agent. While this was not a strict interpretation of the direct perception of affordances from affordance theory it had practical software engineering benefits such as the introduction of the idea of affordances into a multi-agent simulation in a modular and cohesive manner.

On the other hand the Capture The Flag simulation was designed in such a way to allow for multiple interpretations of the computational model of affordance. In one case affordances are computed in the environment and can be directly perceived by the agents, whereas in the other case each agent is responsible for computing (or determining) the affordances available to it in the current simulation time step.

The impact on the design of multi-agent simulations when adopting an affordance based model of agent-environment interaction is discussed in more detail in the subsequent Sections.

7.3 Impact on Design of Multi-Agent Simulations

Many of the broader issues associated with the impact on the design of a multi-agent simulation were discussed in Chapter 4. However this Section deals specifically with lessons learnt with respect to the impact on the design of a multi-agent system when adopting an affordance based approach to agent-environment interaction.

An affordance based approach can impact on the design of a multi-agent simulation in a number of ways. Broadly speaking the design of the agent, the environment and the interaction between the two are the main areas where the design of the simulation is impacted. Based on the experience of developing the CTF and HAVE multi-agent simulations, there are a number factors which can have a significant impact on the design of an affordance based multi-agent simulation. These are:

- How affordances are computed in the multi-agent simulation?
- Where in the simulation architecture affordances are computed?
- When in the simulation how are affordances are computed?
- What affordances are computed?

While these factors can in some regards influence each other, they can also be considered independent and orthogonal to each other. The decisions made by the multi-agent simulation designer as to how, where, when and what affordances are computed in the simulation will have the largest impact on the design. Each of these factors is addressed separately in the following Sections.

7.3.1 How Affordances are Computed

The way in which affordances are determined in a multi-agent simulation will have a significant impact on the design of the system. In other words the single biggest factor that influences the design is the model of affordance being used. In the case of both the HAVE and CTF simulations the interaction model being used was the one presented in Chapter 5.

This model had a number of important features which impacted the design. The first was the dependence on the agent's mental and physical state when determining affordances. For example,, the computation of affordance needed to take into account the agent's current intentions. Furthermore, the model was dynamic, meaning that the affordances were continuously being evaluated every simulation time step.

One can consider a simpler model of affordance in which there is no dependence on the agent's state and capabilities and where the affordances are static throughout the simulation time as described in Chapter 4. In such a case the affordances can be considered as annotated action possibilities on the entities in the environment and which are independent of agent observer. While this type of model is significantly simpler (in that it is not dynamic and does not capture the relational nature of affordances) it nevertheless means that from a design perspective there is less coupling between the agent and environment components in a simulation.

7.3.2 Where Affordances are Computed

Somewhat independent of how affordances are computed is the issue of where in the simulation architecture they are computed. In the CTF simulation two distinct cases were considered; one where affordances were computed in the agent and the other where they were computed in the environment allowing for their direct perception by an agent. The location in the simulation architecture where affordances are computed, can have a significant impact on the design of the system.

The case where affordances are implemented inside an agent means that the addition of affordance based reasoning can be limited to the components of the simulation dealing with agent reasoning. This was the case in HAVE and in the Intelligent Agent variants of the CTF simulation. When dealing with an existing multi-agent simulation, affordances can be introduced into the reasoning processes of an agent without affecting the rest of the simulation architecture.

An agent centred approach has a number of disadvantages. From a modelling perspective it does not capture some of the characteristics of affordances which are considered important in ecological psychology. For example the ability to directly perceive affordances in the environment or the relational nature of affordances.[68]

The largest influence from a design perspective is on the design of any agents in the multi-agent simulation which adopt an affordance based approach. This is because each agent will have to be responsible for the perception of entities in the world, the generation of available affordances, an evaluation of these affordances, deciding which affordance to adopt and finally taking action in the world.

The complexity in the design of an affordance based agent will of course depend on the nature of the affordance model on which the design is based. In the model developed as part of this research, each agent would have to be responsible for computing (or otherwise determining) which affordances are available to it. As shown from the computational run time from the experiments run with the CTF simulation, this can be computationally expensive. This basically means that the design of the individual affordance based agents becomes increasingly complicated and consequently harder to program.

The case where affordances are computed in the environment was implemented in the Intelligent Environment variants of the CTF simulation. In this case aspects of the affordance concept are spread across different parts of the simulation architecture. The environmental model needs to be able to capture all the relations that are required for the computation of affordances (including information from individual agents) as well as determine the affordances for each agent. Agents need to be able to perceive the affordances, evaluate them and decide which ones to adopt. Also the simulation architecture needs be able to provide the appropriate interfacing mechanisms between the agent and the environment to allow the agent to perceive the relevant affordances in the environment.

The main disadvantage of this approach is that the design impact manifests itself across a larger number of subsystems in the simulation architecture. Consequently the introduction

[68]While the relational nature of affordances is captured explicitly in the case where affordances are computed in the environment, this is not the case when affordances are computed by an agent. While the affordance relationships are represented inside the agent they are implicit internal representations, rather than explicit environmental relations.

of affordances not only means that the design of the agent and the environment need to be modified but also the interaction between these modules. Although the design impact is wider across a larger number of subsystems it is not as extensive in those subsystems when compared to the case where the entire agent is redesigned to introduce affordances as in the previous case. This can be an issue when attempting to incorporate affordances into an existing multi-agent simulation as multiple subsystems require changes.

This approach also has a number of advantages. The most obvious one is a closer representation of the concept of an affordance as described in ecological psychology. This type of design can be considered as a higher fidelity representation of the affordance concept. Another significant advantage is that it has the potential to change the way intelligent behaviour is either programmed or incorporated in a multi-agent simulation. In a traditional agent, changes to intelligent behaviour are typically made by the programmer by changing the agent code itself.

However, with this type of approach it is possible to change the behaviour of the agent without changing the agent itself. Rather, the changes in behaviour can be made by making changes to the environment. These changes can range from simple changes to the way entities are annotated to changing the available affordances in the environment so that agents perceive different action possibilities, ultimately changing their resulting behaviour.

With the appropriate tools for creating and modifying virtual environments for use in multi-agent simulations it is possible to make changes to the environment and consequently agent behaviour without making programming changes to any aspects of the simulation.

7.3.3 When to Compute Affordances

In both the CTF and HAVE simulations affordances were determined at simulation run-time. This contributed to the increase in computation time of the simulation. However in some cases it is possible to change when affordances are computed. Instead of computing affordances at run-time they may be pre-computed.

These different approaches to when affordances are computed can have a significant impact on design. Computing at run-time means that the architecture must take into account dynamic information about agents and the environment to determine the available affordances. This provides flexibility at the cost of increased design complexity.

By pre-computing affordances, that is by allowing the agent designer to determine a priori the affordances associated with entities in the world then there is a reduction in the complexity of the design at the cost of decreased flexibility.

The approach to be adopted by a designer of any multi-agent simulation will depend on the requirements of the system being developed and the application domain in question. The decision ultimately is one of trade-offs between performance, design complexity and flexibility.

7.3.4 What Affordances are Computed

The decision as to which affordances should be computed is another factor which impacts on the design of the multi-agent simulation. In the CTF simulation the two major affordance based architectures looked at agent based affordance computation (Intelligent Agent) and environment based affordance computation (Intelligent Environment). However, even with these two major variants the decision is to what affordances should be computed still remained open.

In the most general case affordances for each agent were considered with respect to every single entity in the environment. This approach has the advantage of being completely general and results in an elegant design implementation. However, for a large number of agents and entities, this type of approach is problematic from a computation perspective because of the large number of possible relationships to consider and evaluate.

By reducing the number of affordances being considered and evaluated, it is possible to significantly improve on the computational performance of the simulation. This was the approach taken in the Intelligent Agent II and Intelligent Environment II architectures in the CTF simulation. The number of affordances being evaluated was reduced by taking into account domain specific information.

This was done by partitioning the possible affordances into groups associated with specific entities. For example, only the action possibilities that could be undertaken with respect to a flag on the game map were considered with respect to flags. Similarly actions that could only be taken with respect to enemy tanks were not considered with respect to friendly tanks. By making these groupings (effectively taking into account information about the type of the entity as well as information about social relationships) it was possible to reduce the number of affordance evaluations that were being performed with respect to each agent in each time step.

This type of approach has a number of consequences. First, it introduces a number of special cases that need to be handled and hence the resulting design and implementation is not as elegant and as straight forward when compared to the general case. Second, it makes assumptions about what affordances are available with respect to different entities and hence reduces the possibility that an agent may perceive an affordance with respect to an entity that had otherwise not being explicitly planned for by the designer of the multi-agent simulation.

Ultimately, the decision as to what affordances to consider comes down to making trade-offs with regards to simulation design, computational performance and ease of developing intelligent behaviours. In the next Section a discussion of some of the issues relating to the impact on computational performance, particularly with respect to the CTF simulation, is discussed.

7.4 Impact on Computational Performance

Based on the experiments comparing the computational performance of the various architectures in the CTF multi-agent simulations, it is possible to draw a number of conclusions.

Since the computation of affordances is about presenting the available possible actions to an agent, it is to be expected that this can potentially be a computationally expensive process. Therefore, as expected the computational overhead for a subsystem which computes the affordances for agents will be significantly greater than when simpler agent reasoning approaches such as the one used in the Traditional Agent CTF architecture.

In the CTF simulation, this was the case in the Intelligent Environment architectures where the *Environment* subsystem made the greatest contribution to the computational run time, and also the case in the Intelligent Agent architectures where the *Agent* subsystem was responsible for the most significant amount of computation time.

One of the most important lessons to draw with respect to computational performance is the impact that a naive implementation of an affordance generation algorithm can have on total computation time. In the case of architectures *Intelligent Environment I* and *Intelligent Agent I*, all possible affordances were evaluated for each agent-entity pair. This increased the total computation time significantly.

In comparison, in architectures *Intelligent Environment II* and *Intelligent Agent II*, domain knowledge was taken into account to reduce the number of affordance evaluations. For example, when the algorithm was trying to determine if a set of affordances were available with respect to a flag, only the affordances relevant to flags were evaluated. This more intelligent approach significantly reduced the amount of computation time by reducing the number of affordance evaluations, while keeping the behaviour of the tank agents and the outcomes of the game the same. The disadvantage to this approach was that the implementation of the affordance generation algorithm was less general.

An important lesson in terms of computational performance with respect to the CTF simulation is that it really does not matter if affordances are computed in the environment or in the agent. In the case of *Intelligent Environment II* and *Intelligent Agent II*, the overall average computation time was similar (with the *Intelligent Agent II* case performing slightly better due to the fact that affordances were only evaluated with respect to entities that were visible to each agent, instead of all other entities in the world).

The results also show that it is possible to implement an affordance based approach which does not suffer a significant computational penalty when compared to a traditional agent approach. However, it is also important to note that in the case of the CTF game, the affordance based approaches implemented a more sophisticated and higher fidelity model of agent-environment interaction and agent reasoning.

Ultimately what this means is that the CTF simulation and associated experiment provide evidence to show that when comparing whether an environment or agent focused approach to modelling affordances is being considered, computation time should not necessarily be an overriding factor in making a decision. It is possible to design a multi-agent simulation that models affordance based interaction without computation time being a show stopper. Rather, other considerations such modelling expressivity, model fidelity and other software engineering issues should be considered.

7.5 Impact on Application Domain

In this Section the impact on the application domain of adopting an affordance based approach to multi-agent simulation is discussed. The application domain of military simulation and in particular military operations research is used as an example because it is the domain that motivated this research. [69]

When considering the use of multi-agent simulations in military operations research there are a number of stake-holders. The nature of the simulation environment being used for operations research studies has an impact on these stake-holders. The three most important stake-holders will be considered; namely the operations analysts, the simulation developers and the military subject matter experts.

Operations Research Analysts define the study to be undertaken and use multi-agent simulations to specify scenarios, run experiments and process results in order to answer a set questions and ultimately provide credible scientific advice on military operations and systems.

Simulation Developers are responsible for the specification, design, testing and deployment of the multi-agent simulations used in the studies conducted by the operations research analysts. The developers include the designers of the multi-agent simulation as well as agent and environment designers and programmers.

Military Subject Matter Experts can undertake a number of roles during an operations research study. These roles include working with the operations analysts to define and specify study questions and parameters, receiving the results of the analysis and studies (in this case acting in the role of the client), providing advice and expertise to analysts and developers on military systems and operations and in many cases actively taking part in studies and experiments where they are themselves users of the multi-agent simulations being used.

Adopting an affordance based approach to a multi-agent simulation has the potential to impact each of these stake-holders in different ways.

Analysts and military subject matter experts are required to evaluate and understand complex tactical behaviours that have been modelled in multi-agent simulations. An affordance based approach provides a unique insight into the decisions the agents make in the simulation. It allows for tactical behaviour to be explained in terms of the interaction the agent has with its environment and consequently the possible actions that the agent can take in that environment.

Using the language of affordances allows for military subject matter experts to help verify and validate tactical behaviours because they can directly be related to the action possibilities that they themselves experience in real missions.

Affordance based multi-agent simulations are by no means in wide-spread use and therefore it is difficult to predict if or how they will be used in practice and the impact they will have on

[69]There is no reason why a similar discussion cannot be had using other application domains as examples, such as the development of character intelligent behaviour in video games.

their users. In fact, multi-agent simulations still fill a niche in the wider military simulation community. However, there is some anecdotal evidence to suggest the benefits of using folk psychological agent models such as the BDI model in agent programming languages in military multi-agent simulations [71]. Describing tactical behaviour of intelligent agents in terms of mental attitudes such as beliefs and intentions, has provided a way for analysts, agent programmers and military subject experts to communicate using a common language.

Affordance as mechanism for describing agent-environment interaction has the potential to play a similar role in helping to explain, evaluate and understand situated agent behaviour specifically with relation to a complex environment such as the modern battlespace.

How an affordance based approach impacts on the simulation developer will depend on how affordance based interaction is implemented in a multi-agent simulation. The most significant impact will probably be faced by agent programmers who will need to design intelligent behaviours, taking into account the possible actions that are available to the agent.

While this is still an agent-oriented approach it is a slightly different mind set that the programmer has to be in when compared to purely mental based models of agent reasoning such as the BDI model. The affordance approach means that the agent programmer has to be significantly more aware of the nature of, and interaction with, the virtual environment.

Since agent behaviour then arises from the interaction between the agent and the environment, the design and programming of the virtual environment becomes as important as the design of the agents in the simulation. If the action possibilities being considered in a multi-agent simulation include aspects of the environment that go beyond the physical, then the more abstract aspects of the environment (such as the social and organisational aspects) must be considered. This means that an approach capturing the multi-modal nature of the agent-environment interaction needs to be considered.

Ultimately, one could consider a virtual environment where significant changes to tactical behaviours are made simply by changing the annotations (such as affordances) in a virtual environment. With the appropriate tools this has the potential to minimise the amount of agent programming required and allow an operations analyst to change the nature of agent intelligent behaviour by changing the definition and specification of the virtual environment.

Perhaps the most important lesson learnt from the use of an affordance based approach to agent reasoning in the CTF and HAVE multi-agent simulations, is the different mind set that both designers and users of agents must come to terms with, when thinking about intelligent behaviour in the context of action possibilities.

A common approach to designing intelligent agents especially in video games is to make use of finite state machines (FSMs) [118, 43]. In this approach the agent designer must think about all the possible states that the agent could be in, as well as how to transition from one state to the other.

Implementation of FSMs involves programming the behaviours for each individual state as well as defining the rules and conditions under which state transitions can occur. Variations on this basic approach include hierarchical finite state machines where each state can itself contain multiple state machines as well as fuzzy finite state machines where a transition from one state to another may include a probabilistic element.

Agent designers and programmers using the BDI reasoning model on the other hand need to think about the agent behaviour and the specific problem being solved in terms of constructs representing mental attitudes such beliefs, desires and intentions. In plan based BDI agent programming languages (such as dMARS and JACK), this involves the agent designer not only thinking about the beliefs the agent can have in a specific situation, but also the plans as executing intentions (or recipes) for achieving specific goals.

When developing the affordance based agents in both HAVE and CTF, it was found that a different programming mind set was required when compared to programming a traditional agent oriented system.

The most important difference is one of perspective. Whereas in finite state machine or BDI agents the focus is in on the internal state of the agent, an affordance based approach forces the agent designer to bring the interaction between the agent and the environment to the forefront. Designing intelligent behaviour becomes an issue of determining what possible actions are available to an agent in different situations. An affordance becomes the primary driver of intelligent behaviour, emphasising the importance of the interaction with the environment.

An affordance based perspective is not only useful when implementing an affordance based agent (at coding time), but also at specification and design time – independent of the implementation mechanism. This is important not only for those involved in the engineering of a multi-agent simulation but also for the users. In a military simulation system such as HAVE, the users of the system are the operations analysts tasked to conduct studies and experiments as well as military operators such as pilots who act as subject matter experts (SMEs) or as experimental participants.

Specifying and explaining agent behaviour in terms of the interactions the agents have with the environment is beneficial for both the simulation engineers and the users of the simulation. The terminology and language associated with affordance oriented interaction provides a common language that facilitates discourse on agent behaviour in a similar way to representations of mental attitudes in BDI agent programming languages.

7.6 Alternative Directions

The model of affordance oriented agent – environment interaction described in Chapter 5 was focused on describing the interaction between a single agent and a single entity in the environment. However, the model (and ultimately the resulting implementations) could have taken on a number of different directions. In this Section, a number of these alternative directions are explored. These can be seen as possible ways in which the affordance based interaction model could be extended in the future.

By making the concept of an affordance somewhat more abstract, there are a number of ways of extending this agent-entity model of affordance that may prove useful to practical multi-agent simulations.

The consideration of affordances has been with respect to entities that agents can perceive in the environment. However, one might consider affordances with respect to entities that are not currently visible to the agent, but (in the language of the BDI model) the agent has beliefs about. For example, if an agent can no longer perceive an entity in the world because for whatever reason it has been obscured from view, that does not necessarily mean that the agent still does not maintain beliefs (and therefore has some knowledge) of the entity in question. Therefore, it is plausible to consider the case that an agent might be able to reason about action possibilities with respect to an entity that was once recently visible, or has some a priori knowledge about.

Affordances with respect to beliefs about entities has a number of implications.

- Allows for agents to reason about affordances with respect to entities which are no longer detected by the agent's sensors.
- Allows for agents to reason about affordances with respect to entities that they have yet to perceive with their sensors but believe to exist.[70]
- Allows for agents to make mistakes when reasoning about affordances with respect to a belief they have about an entity which may vary significantly from the actual properties of an entity.
- Affordances with respect to beliefs about entities can not be easily reconciled with the view prevalent in ecological psychology that affordances are directly perceivable in the environment.
- Considering affordances from the perspective of the agent (as internal mental constructs) makes it easier to implement this concept in a multi-agent simulation than in the case when affordances are represented as a concept directly perceivable in the environment.

Affordances with respect to entities not currently visible were used in the Capture The Flag simulation with respect to friendly and enemy bases. The tank driver agents knew that the enemy base afforded being attacked while the friendly base afforded being defended, even if the bases were not initially within the field of view of the agent's sensors.

Another possible extension to the model is to consider affordances with respect to relations in the world. For example, it is possible to perceive affordances with respect to things that do not have an actual physical manifestation in the world. A gap or space formed by a door frame or the relative position of two large rocks may afford passing through. The gap or space is not an entity that physically manifests itself but can still be perceived by an agent. This is consistent with the view in ecological psychology that relations in the world exist and hence are perceivable.

The idea can be taken further to consider not only affordances with respect to physical relations, but affordances with respect to other abstract relations in the environment, such as social and organisational relations. Furthermore, one might also consider affordances with respect to relations that are not directly perceivable. That is, an agent might be able to reason about affordances with respect to beliefs about certain relations. For example, in a military context, if one agent believes that another agent is its commander then there are a number of action possibilities that can arise due to the belief about that relationship.

[70] In a military simulation this can include agents which have been pre-briefed (possibly relying on information from intelligence sources) about particular entities such as targets.

Tables 7.1: Affordances, afforders and affordees: Columns represent the two types of *intelligent* entities that can perceive affordances – agents and teams of agents. The rows represent the types objects in the world which agents and teams can have affordances with respect to. These include entities (which may include standard entities, agents and teams), relations and situations. The most general case is represent by the affordance relation $\phi^k(\tau, \sigma)$ which represents the k^{th} affordance, afforded to team τ by situation σ.

	Agent	Team
Entity	$\phi^k(\alpha, \varepsilon)$	$\phi^k(\tau, \varepsilon)$
Relation	$\phi^k(\alpha, \rho)$	$\phi^k(\tau, \rho)$
Situation	$\phi^k(\alpha, \sigma)$	$\phi^k(\tau, \sigma)$

By abstracting further one may consider that possible actions afforded by a situation. That is, instead of considering the case of the affordances provided by entities or relations to specific agents, one can consider the affordances provided by a particular situation in which an agent finds itself.

In addition to the affordances perceived by agents, one can consider the action possibilities provided to groups of agents such as teams or organisations. This is particularly relevant in multi-agent military simulation such as air combat modelling where fighter aircraft typically operate in teams of two or four. One can then consider the affordances provided to a team of agents with respect to a particular entity in the world, and also the affordances a team of agents can have in a particular situation.

Affordances for agents can be considered as a special case of team oriented affordances if Tidhar's model of team and organisational oriented systems is adopted [155], where an agent is a team with one member. The idea of team oriented affordances has not been considered in ecological psychology and hence departs from the mainstream theory of affordance.

In addition to the base case of affordances between agents and entities, Table 7.1 summarises the different types of affordance relationships that can be considered by extending the model. The table shows that the two types of afforders (entities which perceive affordances) are agents and teams while there are three types of afforders (entities, relations and situations).

7.7 Summary

This purpose of this Chapter was to bring together and discuss some of the issues that were raised from the development of the affordance model and its subsequent implementations in the CTF and HAVE architectures.

The Chapter began by discussing some of the challenges of modelling and simulating affordances in a multi-agent simulation. This was followed by a discussion of the impact that adopting an affordance based approach has on a multi-agent simulation. The three areas that were looked at were (i) the impact the adoption has on the design, (ii) the computational performance and (iii) the use of the simulation in the relevant application domain.

The Chapter finishes with a brief discussion of possible alternative directions which the affordance model presented in this thesis could have taken, as well as a brief outline of some of the future directions and extensions that can be made to this work.

Chapter 8

Conclusion

Dwight Schrute: "Second Life is not a game. It is a multi-user, virtual environment. It doesn't have points or scores. It doesn't have winners or losers."
Jim Halpert: "Oh, it has losers."
— The Office (Season 4, Episode 5)

8.1 Review of Thesis

This thesis looked at the problem of using the concept of affordances from ecological psychology to model the interaction between computational intelligent agents and the virtual environments in which they are situated in the context of multi-agent simulations.

An affordance based model of agent-environment interaction was developed and described in Chapter 5. The model was then implemented in two practical multi-agent simulations. The first, described in Chapter 2 called the Human Agent Virtual Environment (HAVE) was a complex military multi-agent simulation. The second, was a multi-agent simulation of the game Capture The Flag (CTF).

By developing these multi-agent simulations it was possible to evaluate the viability of an affordance based approach to agent-environment interaction. In particular, the Capture The Flag multi-agent simulation was used to empirically evaluate different approaches to implementing an affordance oriented approach – one where affordances are computed in the agent and the other where affordances are computed in the environment.

A review of the most significant aspects of each of the proceeding Chapters of this thesis is now presented.

Chapter 1
Introduction

Chapter 1 described the problem being addressed by this thesis – modelling agent – environment interaction in multi-agent simulations with affordances. It included a definition of affordances that was used in this work and described how affordance theory was used to inspire designs for agent – environment interaction in multi-agent simulations.

The Chapter described how existing approaches for designing intelligent behaviour typically involved one of two approaches – either designing a more intelligent agent or designing a more intelligent environment.

It was explained that in this thesis, a third approach was being adopted, one where the focus is on the interaction between agent and environment. As such, affordance theory from ecological psychology was well suited to not only model this interaction but could also be used to address issues of simulation complexity management and possibly simulation computational performance.

The Chapter then described the work in military multi-agent simulation which motivated this research – in particular multi-agent simulation of air operations used for the purposes of operations research.

The contributions made by this research were then described, the most significant ones being the development of an affordance model of interaction and analysis of an affordance based approach in agent – environment interaction in multi-agent simulations. The contributions made by this research are revisited in Section 8.2.

Chapter 2
Motivation: The Human Agent Virtual Environment

Chapter 2 described the Human-Agent Virtual Environment, a military operations research simulation of a Close Air Support mission that implements a version of the affordance based interaction model described in Chapter 5. The implementation of HAVE demonstrated the feasibility of not only an affordance based reasoning model, but also a multi-modal representation of the virtual environment in a multi-agent simulation of a complex real world application domain.

While the affordance model implemented in the HAVE simulation was described later in the thesis (Chapter 5) the description of HAVE was presented earlier in the thesis because it is a motivating example of the type of complex multi-agent simulations that can benefit from an affordance oriented approach to modelling the agent – environment interaction.

Chapter 3
Literature Review: Situating Agents in Virtual Environments

The focus of Chapter 3 was to review the relevant literature relating to situating computational intelligent agents in virtual environments. The review looks at the areas which have influenced the development of the affordance based model. This includes work in the field of situated cognition, the beliefs, desires and intentions (BDI) model of agent reasoning and Boyd's OODA loop model of military decision making.

The Chapter also includes a review of related work where affordances or affordance like constructs have been used in the fields of multi-agent simulation, intelligent virtual environments and interactive entertainment.

A review of the relevant literature on affordance theory from ecological psychology is found in Chapter 5.

Chapter 4
Background: Affordance Oriented Interaction

Chapter 4 looked at the broader issues associated with developing an affordance based model of agent-environment interaction.

The Chapter focused on three particular areas. The first looked at affordances as annotations that could be made on entities in the virtual environment. The second looked at the types of environmental representation issues that needed to be addressed if an affordance based approach was to be adopted in multi-agent simulation – the most important of which, were the need for an explicit model of the virtual environment and the desire for a multi-modal representation of the environment. The third looked at some of the issues associated with agent representation if an affordance oriented approach was adopted.

Chapter 5
Modelling Affordances

This Chapter first began by describing the aspects of the affordance concept from the ecological psychology literature that were relevant to multi-agent simulation. The Chapter then presented a list of concepts that needed to be taken into consideration to adequately model the concept of an affordance. The concepts were put together to describe a model of affordance which was presented as a relation between an agent and another entity in the environment. This included a formal model that was described in the Z specification language and incorporated aspects of Boyd's Observe-Orient-Decide-Act (OODA) loop model of military decision making. A number of illustrative examples from the game Capture The Flag were also described.

Chapter 6
Evaluation: Capture The Flag

In Chapter 6 the implementation and associated experiments of a multi-agent simulation environment of the game Capture The Flag (CTF) was described. This was done in order to explore the impact of different design and implementation choices for the affordance based model of agent-environment interaction.

In particular, two extreme cases were compared. In one case affordances were computed inside the agent, treated like any other mental or cognitive construct. In the other case, affordances were computed in the environment allowing agents to directly perceive affordances in the world.

The two design and implementations (as well as a number of variants) allowed for a comparison to be made of the different approaches from the perspective of design, computational performance, and model fidelity.

Chapter 7
Discussion

Chapter 7 brought together collective insights and lessons in a general discussion about the various aspects of modelling affordances and implementing them in multi-agent simulations. The Chapter focused on the impact that an affordances based approach had on the design of a multi-agent simulation, the impact on computational performance and most importantly the impact it has on the application domain in which the simulation is actually being used.

8.2 Contributions

This thesis has demonstrated the viability of an affordance based approach to modelling the interaction between agents and their environments in multi-agent simulations. It has shown that an affordance based approach to modelling the interaction can potentially lead to agents that are more situated and have an intrinsically richer interaction with their environment.

The primary contributions from this research are:

- An affordance model of agent – environment interaction.
- An analysis of the impact of adopting an affordance based approach in multi-agent simulations.

The affordance based model of interaction is a novel and original contribution for a number of reasons. First it captures important properties of affordance which have not been fully addressed or considered in previous work. Second, the model is framed in terms of concepts from the BDI model of agent reasoning and Boyd's OODA loop model of military decision making. This framing shows that the model is not incompatible with existing approaches to agent reasoning and can be incorporated into existing multi-agent simulations. Third, the model is specified in an abstract manner to allow for different design interpretations. That is the model allows for the computation of affordances to be computed in different parts of the multi-agent simulation - whether they be in the agent, in the environment or in the interface between the two.

The implementation of the affordance based model in two multi-agent simulations (HAVE and CTF), allowed for the analysis of the impact of taking an affordance based approach. Specifically the analysis looks at the impact on three specific areas; the impact on the design of the multi-agent simulation, the impact on computational performance and the impact on the application domain making use of the simulation.

The thesis also made a number of smaller contributions in terms of lessons learnt and insights gained from the development of the affordance oriented interaction model and the two multi-agent simulations. These include:

- Demonstrating the importance of an explicit and a multi-modal representation of the virtual environment when adopting an affordance oriented approach in a multi-agent simulation.

- Demonstrating that it is possible to implement an affordance based approach in a multi-agent simulation in a number of different ways. This included approaches where affordances were computed in the environment and where affordances were computed in the agent. The differences in terms of impact on the design and computational performance of the multi-agent simulation when adopting one of these approaches was explained.
- Demonstrating how an affordance based approach can be incorporated into an existing multi-agent simulation.
- Describing how affordances in simulations can be formulated in terms of special types of annotations in the virtual environment.
- Describing how computational performance of a general approach to affordance computation can be improved by taking into account domain specific information in a more specific affordance generation algorithm.
- Demonstrated a systematic way of capturing and modelling the complex interactions between simulation entities and the environment based on an affordance model.

8.3 Applications of Modelling Affordances

While the focus in this thesis was modelling affordances as a mechanism for agent-environment interaction in multi-agent simulations, the potential exists to apply this work to a number of application domains.

The principle application domain which motivated this research is in the use of affordance based interaction in constructive multi-agent simulations used in military operations research. While the focus has been on military operations research simulation it is also applicable in multi-agent simulations where there is human involvement, such as in training simulators and in military experimentation.

However, as discussed in Chapter 3 there is significant potential for the use of affordance based approaches in other types of multi-agent based simulation systems, the most promising being in the area of video games. There are many similarities between many modern video game genres and current military simulation technology.

This means that many of the ideas discussed in this thesis in the context of military simulation can be easily applied to the development of intelligent character behaviour in video games. The use of intelligent agents to represent complex decision making processes (whether they represent game characters or military operators) as well as the use of complex virtual environments are just two of the many similarities between the domains which make many of the ideas regarding modelling affordances that were presented in this thesis equally applicable to the video game domain.

In the long term, modelling affordances computationally may also prove beneficial to the ecological psychology and wider cognitive science communities as they hope to gain greater insight into the concept of an affordance and a better understanding of how humans and animals interact with their environment.

8.4 Limitations

The research presented in this thesis has a number of limitations where there is either room for improvement, potential for different approaches being tried as well as the opportunity for future work through the form of extensions.

The limitations generally fall into two categories. These are limitations in the theoretical aspects of this research which largely revolve around the development of the affordance based model of agent-environment interaction, and limitations in the practical implementation of the affordance model in the two example systems and the associated experiments. The main limitations of this work can be summarised as follows.

8.4.1 Wider Applicability

The affordance based model of agent-environment interaction that was developed was heavily influenced by experience in the development of multi-agent simulations for the purposes of military operations research. Hence it is a model that has been developed with a particular mind-set and within the context of a particular application domain. Therefore this model considered aspects from the theory of affordances that were relevant to this experience.

The model described in this thesis is but one of many possible models of agent-environment interaction inspired by the theory of affordance that could have been developed. One can argue about how widely this model can be applied. The question of wider applicability has not been explored in this thesis.

8.4.2 Model Fidelity

A potential limitation is the fidelity of the model. A balance had to be reached between making the model more sophisticated and its use in practical, faster than real time multi-agent simulations. A potential criticism from ecological psychologists is that the model does not adequately capture sufficiently important aspects of affordance theory and hence the constructs in the various implementations cannot be actually called affordances.

It is important to note however, that it was not the intention to produce a valid model of affordance from an ecological psychology perspective. Rather the aim was to develop a model of agent-environment interaction that was inspired and motivated by affordance theory.

The question of model fidelity should however be requirements driven. Even in the case of military operations research there will be particular problems where the model presented in this thesis would be considered too detailed, while in other cases it might not be detailed enough.

8.4.3 Model Generalisation

The experiments conducted with respect to the CTF multi-agent simulation were primarily focused on the total computation time associated with various implementations of the affordance based interaction model.

A number of different areas could have been further explored and compared. These include a more detail design analysis, such as using traditional software engineering metrics to compare design complexity, as well as the ease or otherwise of programming intelligent behaviours using an affordance based approach from a programmers perspective.

Perhaps the most significant limitation in this area is a lack of detailed comparison with existing agent programming languages; although this would have significantly increased the scope of the thesis.

8.5 Future Work

There are many areas in which this research can be extended. However there are four specific areas which at this stage show significant promise. These include:

- Exploring team and situation oriented affordance models.
- Increasing the number of tanks and other entities in the game (as well as including newer entities such as obstacles) thereby increasing the complexity of the environment in which the tank agents need to operate.
- An increase in the number of tanks on each side would also allow the exploration of affordance models with respect to more sophisticated social and organisational structures (such as sub-teams).
- Exploring additional affordance based architectures. Some possibilities include hybrid architectures both at the agent-environment interaction level and at the agent reasoning level. For example, one possibility may include extending existing agent programming languages to introduce affordances as first order programming constructs.

While the research presented in this thesis has focused on modelling agent – environment interaction with affordances in multi-agent simulation, it can be considered a step towards a larger, long term research challenge and aspiration.

There is an increasing requirement for agents and humans to interact with each other in virtual environments in a manner which is authentic, plausible and on an equal footing. Typically in military simulations this takes on the form of agents acting as adversaries to humans participants in a simulation. However, there is also a need for humans and agents to interact, co-operate and coordinate as members of the same team in a single virtual environment.

The need for credible hybrid human-agent teams poses many long term research challenges. One such challenge is to ensure that both humans and agents, whether they be on the same or opposing teams, have access to the same type of perceptual information in the virtual environment. Given that ecological psychology suggests that humans can perceive affordances (or action possibilities) it is sensible to consider that the intelligent agents in the world can perceive affordances as well. In this manner it is possible to consider affordances as a common language which describe how humans and agents perceive action possibilities in the virtual environment.

The Human Agent Virtual Environment (HAVE) was an affordance based military multi-agent simulation developed as a first step towards this aspirational long term goal. This simulation system is described in Chapter 2.

While there are many avenues for expanding this work in the future, there are a number of areas which are identified here because they have the potential to drive the concept of affordance oriented agent – environment interaction in interesting directions.

Further Development of Affordance Model: The model of affordance described in this thesis was largely influenced by the practical considerations associated with developing multi-agent simulations. However there is potential for the model to be developed in the future to incorporate a large range of concepts from the fields of ecological psychology and the wider situated cognition community. The continual development and refinement of the affordance concept in the ecological psychology community provides an opportunity for these ideas to be incorporated into affordance based interaction models for multi-agent systems.

The model of affordance described in Chapter 5 incorporated ideas from on the nature of affordance from a number of sources including the work of Gibson [54], Norman [103, 104], Clancey [27], Jones [78], Stoffregen [143, 142], Kirlik [84], Michaels [97], Heft [61] and Chemero [23, 24]. [71] However there is scope to incorporate additional concepts from these sources into a future model of affordance.

Similarly work done in the field of situated action [146, 101, 1, 25, 59, 145, 162, 164, 163] has the potential to influence the direction of any future models of affordance for multi-agent simulation. While not dealt with in an in depth manner in this thesis, the field of situated action is relevant to understanding affordances. [72]

Additional Application Domains: The concept of an agent-environment interaction based on affordances needs to be explored beyond the military simulation (HAVE) and video game (CTF) domains. By investigating additional domains (both other multi-agent simulations and other applications of intelligent agents) it is possible to build a better picture of how useful the concept of affordance is to agent-environment interaction. It was claimed earlier that many agent applications and programming languages largely ignored the interaction with the environment. The research presented in this thesis was one attempt to try to address this problem through the concept of an affordance. In the application domains presented it was shown to be a

[71] A number of special issues of the Journal of Ecological Psychology in which some of these papers appear are relevant to pursuing further work on the development of the affordance model. Specifically the special issues titled *How Are Affordances Related to Events? An Exchange of Views* [89] and *How Shall Affordances Be Refined? Four Perspectives* [77] are good starting points.

[72] Most of the papers referenced here can be found in a 1993 special issue of the Journal of Cognitive Science titled *Cognition in the Head and in the World: Special Issue on Situated Action* [102].

viable and useful mechanism for modelling the agent-environment interaction. The question however still remains if the concept can be applied more widely into many other agent application domains.

Team Oriented Affordances: Chapter 5 described how the standard affordance based interaction model could be extended to consider affordances with respect to teams and particular situations. There is potential for significant research to be undertaken in this area not only from the perspective of multi-agent systems but also from the perspective of ecological psychology, where the idea of social and organisational structures such as teams perceiving affordances in the world is unexplored.

Hybrid Affordance Architectures: The affordance model described in this thesis was largely stand-alone. While it borrowed some ideas from existing agent models such as the BDI model and Boyd's OODA loop model, there is the potential for the development of a hybrid agent architecture or model which incorporates the idea of an affordance into an existing agent model such as the BDI model.

Affordance as a Programming Construct: Following on from the idea of a hybrid agent affordance architecture, it makes sense to ask the question of whether there is utility in the concept of an affordance becoming a first order agent programming languages construct in the same way that beliefs, goals and plans are in BDI based programming languages.

References

1. Philip E. Agre. The symbolic worldview: Reply to Vera and Simon. *Cognitive Science*, 17(1):61–69, 1993.

2. Ebrahim Al-Hashel, Bala M. Balanchandran, and Dharmendra Sharma. A comparison of three agent-oriented software development methodologies: Roadmap, prometheus, and mase. In *Knowledge-Based Intelligent Information and Engineering Systems*, volume 4694 of *Lecture Notes in Computer Science (LNCS)*, pages 909–916, Heidelberg, Germany, September 2008. Springer.

3. R. St. Amant. User interface affordances in a planning representation. *Journal of Human Computer Interaction*, 14(3):317–354, 1999.

4. Michael L. Anderson. Embodied cognition: A field guide. *Artificial Intelligence*, 149:91–130, 2003.

5. R. Arkin, F. Cervantes-P'erez, and A. Weitzenfeld. Ecological robotics: A schema-theoretic approach. In R C Bolles, H Bunke, and H Noltemeier, editors, *Intelligent Robots: Sensing, Modelling and Planning*, volume 27 of *Series in Machine Perception and Artificial Intelligence*, page 476. World Scientific, 1997.

6. M. Atkin, D. Westbrook, and P. Cohen. Capture the flag: Military simulation meets computer games. In *AAAI Spring Symposium on AI and Computer Games*, 1999.

7. Ronald T. Azuma. A survey of augmented reality. *Presence: Teleoperators and Virtual Environments*, 6(4):355–386, August 1997.

8. José-Antonio Báez-Barranco, Tiberiu Stratulat, and Jacques Ferber. A unified model for physical and social environments. In Danny Weyns, H. Van Dyke Parunak, and Michel Fabien, editors, *Environments for Multi-Agent Systems III: Proceedings of the Third International Workshop on Environments for Multiagent Systems (E4MAS 2006)*, volume 4389 of *Lecture Notes in Computer Science (LNCS)*, pages 41–50, Hakodate, Japan, May 2007. Springer-Verlag, Berlin, Germany.

9. Robert E. Ball. *The Fundamentals of Aircraft Combat Survivability Analysis and Design.* AIAA Education Series. American Insitute of Aeronautics and Astronautics, Reston, Virginia, USA, 2003.

10. Stefania Bandini, Sara Manzoni, and Carla Simone. Dealing with space in multi-agent systems: a model for situated mas. In *The First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2002)*, pages 1183–1190, Bologna, Italy, July 15-19 2002. ACM.

11. Stefania Bandini, Sara Manzoni, and Giuseppe Vizzari. A spatially dependent communication model for ubiquitous systems. In Danny Weyns, H. Van Dyke Parunak, and Fabien Michel, editors, *Environments for Multi-Agent Systems (E4MAS), Lecture Notes in Computer Science (LNCS)*, volume 3830, pages 74–90. Springer, 2005.

12. Stefania Bandini and Giuseppe Vizzari. Regulation function of the environment in agent-based simulation. In Danny Weyns, H. Van Dyke Parunak, and Michel Fabien, editors, *Environments for Multi-Agent Systems III: Proceedings of the Third International Workshop on Environments for Multiagent Systems (E4MAS 2006)*, volume 4389 of *Lecture Notes in Computer Science (LNCS)*, pages 157–169, Hakodate, Japan, May 2007. Springer-Verlag, Berlin, Germany.

13. Jon Berndt. JSBSim - an open source flight dynamics model in C++. In *AIAA Modeling and Simulation Technologies Conference and Exhibit*, Providence, RI, USA, 2004. AIAA.

14. Carole Bernon, Massimo Cossentino, and Juan Pavi. An overview of current trends in European AOSE research. *Informatica*, 29(5):379–390, November 2005.

15. Jeffrey Bradshaw. *Software Agents.* MIT Press, 1997.

16. Frances M. T. Brazier, Barbara Dunin-Keplicz, Jan Treuer, and Rineke Verbrugge. Modelling internal dynamic behaviour of BDI agents. In *ModelAge Workshop*, pages 35–56, 1997.

17. Rodney A. Brooks. Intelligence without representation. *Artificial Intelligence Journal*, 47:139–159, 1991.

18. Vicki Bruce and Patrick Green. *Visual Perception: Physiology, Psychology and Ecology.* Lawrence Erlbaum and Associates, Hove, East Sussex, UK, 2nd edition edition, 1990.

19. Don Burns and Robert Osfield. Open Scene Graph a: Introduction, b: Examples and applications. In *VR '04: Proceedings of the IEEE Virtual Reality 2004 (VR'04)*, page 265, Washington, DC, USA, 2004. IEEE Computer Society.

20. Chris Butcher and Jaime Griesemer. The illusion of intelligence: The integration of AI and level design in Halo. In *Proceedings of the 2002 Game Developers Conference (GDC 2002)*, San Jose, California, March 2002.

21. Air Power Development Centre. *The Air Power Manual.* Royal Australian Air Force, 5th edition edition, 2007.

22. Alex J. Champandard. Living with the Sims' AI: 21 tricks to adopt for your game. *http://aigamedev.com/reviews/the-sims-ai*, October 15th 2007.

23. Anthony Chemero. What we perceive when we perceive affordances: Commentary on Michaels (2000) "information, perception, and action". *Journal of Ecological Psychology*, 13(2):111–116, 2001.

24. Anthony Chemero. An outline of a theory of affordances. *Journal of Ecological Psychology*, 15(2):181–195, 2003.

25. William J. Clancey. Situatedaction: A neuropsychological interpretation response to Vera and Simon. *Cognitive Science*, 17(1):87–116, 1993.

26. William J. Clancey. Chapter 11: The ecological approach to perception. In *Situated Cognition: On Human Knowledge and Computer Representation*, pages 243–265. Cambridge University Press, Cambridge, New York, Melbourne, 1997.

27. William J. Clancey. *Situated Cognition: On Human Knowledge and Computer Representations.* Cambridge University Press, Cambridge, New York, Melbourne, 1997.

28. David J. Cloud and Larry B. Rainey, editors. *Applied Modeling and Simulation: An Integrated Approach to Development and Operation.* Space Technology Series. McGraw-Hill, 1998.

29. Robert Coram. *Boyd: The Fighter Pilot Who Changed the Art of War.* Little, Brown and Company, Boston, New York, London, first edition, 2002.

30. Jason B. Cornwell, Kevin O'Brien, Barry G. Silverman, and Josef A. Toth. Affordance theory for improving the rapid generation, composability and reusability of synthetic agents and objects. In *Twelfth Conference on Computer Generated Forces and Behavior Representation*, 2003.

31. Kerstin Dautenhahn. Story-telling in virtual environments. In *Intelligent Virtual Agents Workshop, 13th Biennial European Conference on Artificial Intelligence (ECAI-98)*, Brighton, UK, 1998.

32. Daniel Dennett. *Kinds of Minds. Toward an Understanding of Consciousness.* Basic Books, New York, 1996.

33. Mark d'Inverno, David Kinny, Michael Luck, and Michael Wooldridge. A formal specification of dMARS. In M. P. Singh, A. S. Rao, and M. Wooldridge, editors, *Intelligents Agents IV: Proceedings of the Fourth International Workshop on Agent Theories, Architectures and Languages (ATAL 97)*, volume LNAI 1365, pages 155–176, Providence, Rhode Island, USA, 1997. Springer-Verlag, Berlin.

34. Mark D'Inverno and Michael Luck. *Understanding Agent Systems.* Springer Series on Agent Technology. Springer-Verlag, Berlin, 2001.

35. Mark D'Inverno, Michael Luck, Michael Georgeff, David Kinny, and Michael Wooldridge. The dMARS architecture: A specification of the distributed multi-agent reasoning system. *Autonomous Agents and Multi-Agent Systems*, 9(1-2):5–53, 2004.

36. Patrick Doyle. Virtual intelligence from artificial reality: Building stupid agents in smart environments. In *AAAI '99 Spring Symposium on Artificial Intelligence and Computer Games*, Stanford, California, USA, 1999.

37. Patrick Doyle. Believability through context: Using "knowledge in the world" to create intelligent characters. In *International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 342–349, Bologna, Italy, 2002.

38. Patrick Doyle and Barbara Hayes-Roth. Guided exploration of virtual worlds. Technical Report KSL 97-04, Knowledge Systems Laboratory, Department of Computer Science, Stanford University, May 1997 1997.

39. Patrick Doyle and Barbara Hayes-Roth. Agents in annotated worlds. In Katia P. Sycara and Michael Wooldridge, editors, *Proceedings of the 2nd International Conference on Autonomous Agents (Agents'98)*, pages 173–180, Minneapolis, USA, 1998. ACM Press, New York.

40. Andrew Duchon and William Warren. Robot navigation from a Gibsonian viewpoint. In *IEEE Conference on Systems, Man and Cybernetics, 1994*, San Antonio, Texas, USA, 1994. IEEE.

41. Andrew Duchon, William Warren, and Leslie Pack Kaelbling. Ecological robotics: Controlling behavior with optical flow. In .D. Moore and J.F. Lehman, editors, *Proceedings of the 17th Annual Cognitive Science Conference*, pages 164–169. Lawrence Erlbaum Associates, 1995.

42. Andrew P. Duchon, William H. Warren, and Leslie Pack Kaelbling. Ecological robotics. *Adaptive Behavior*, 6(3):473–507, 1998.

43. Eric Dybsand. A finite-state machine class. In Mark DeLoura, editor, *Game Programming Gems*, chapter 3.1, pages 237 – 248. Charles River Media, Rockland, Massachusetts, USA, 2000.

44. Richard Evans. The future of ai in games. *Game Developer*, 2001.

45. N. Farenc, S. Musse, E. Schweiss, M. Kallmann, O. Aune, R. Boulic, and D. Thalma. One step towards virtual human management for urban environments simulation. In *ECAI'98 Workshop of Intelligent Virtual Environments*, Brighton, UK, 1998.

46. J. Ferber, F. Michel, and J.-A. Báez-Barranco. Agre: Integrating environments with organizations. In D. Weyns, H. Van Dyke Parunak, and F. Michel, editors, *Environments for Multi-Agent Systems : Proceedings of the First International Workshop on Environments for Multi-Agent Systems (E4MAS 2004)*, volume 3374 of *Lecture Notes in Computer Science (LNCS)*, pages 48–56, New York, NY, USA, July 2004. Springer, Berlin.

47. Jacques Ferber. *Multi-Agent Systems.* Addison Wesley, 1999.

48. Richard M. Fujimoto. *Parallel and Distributed Simulation Systems.* John Wiley and Sons, Inc., New York, 2000.

49. David Gelernter. *Mirror Worlds: or the Day Software Puts the Universe in a Shoebox...How It Will Happen and What It Will Mean.* Oxford University Press, 1993.

50. M.P. Georgeff and F. F. Ingrand. Decision-making in an embedded reasoning system. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence (IJCAI-89)*, pages 972–978, Detroit, MI, 1989.

51. M.P. Georgeff and A. L. Lansky. Reactive reasoning and planning. In *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87)*, pages 677–682, Seattle, WA, USA, 1987.

52. Elanor J. Gibson, Karen Adolph, and Marion Eppler. Affordances. In Robert A. Wilson and Frank C. Keil, editors, *The MIT Encyclopedia of the Cognitive Sciences*, pages 4–6. The MIT Press, Cambridge, Massachussetts, 1999.

53. James J. Gibson. The theory of affordances. In R. E. Shaw and J. Bransford, editors, *Perceiving, Acting and Knowing*. Lawrence Erlbaum and Associates, Hillsdale, New Jersey, 1977.

54. James J. Gibson. *The Ecological Approach to Visual Perception*. Resources for Ecological Psychology. Lawrence Erlbaum and Associates, Hillsdale, New Jersey, 1986.

55. Marco Gillies and Neil Dodgson. Invariants and affordances for walking in a cluttered environment. In Daniel Ballin, editor, *Second Workshop on Intelligent Virtual Agents*, University of Salford, UK, 1999.

56. S. Goss, C. Heinze, M. Papasimeon, A. Pearce, and L. Sterling. The importance of being purposive: Towards reuse in agent oriented information systems. In Paolo Giorgini, Brian Henderson-Sellers, and Michael Winikoff, editors, *Proceedings of the Workshop on Agent Oriented Information Systems (AOIS 2003)*, Lecture Notes in Artificial Intelligence (LNAI), pages 110–125, 2004.

57. Simon Goss, editor. *Proceedings of the First International Workshop on Team Tactics and Plan Recognition*, 1999.

58. Abdelkader Gouaich and Fabien Michel. Towards a unified view of the environment(s) within multi-agent systems. *Informatica*, 29(4):423–432, November 2005.

59. James G. Greeno and Joyce L. Moore. Situativity and symbols: Response to Vera and Simon. *Cognitive Science*, 17(1):49–59, 1993.

60. Barbara J. Grosz and Sarit Kraus. Collaborative plans for complex group action. *Artificial Intelligence*, 86(2):269–357, 1996.

61. Harry Heft. Affordances, dynamic experience, and the challenge of reification. *Journal of Ecological Psychology: Special Issue: How are Affordances Related to Events? An Exchange of Views*, 15(2):149–180, 2003.

62. C. Heinze, S. Goss, T. Josefsson, K. Bennett, S. Waugh, I. Lloyd, G Murray, and J. Oldfield. Interchanging agents and humans in military simulation. *AI Magazine*, 23(2):37–47, 2002.

63. C. Heinze, S. Goss, I. Lloyd, and A. Pearce. Collaborating cognitive and sub-cognitive processes for the simulation of human decision making. In *Proceedings of the Third International Simulation Technology and Training Conference (SimTect '98)*, Melbourne, Australia, 1998.

64. C. Heinze, S. Goss, and A. Pearce. Plan recognition in military simulation: Incorporating machine learning with intelligent agents. In *Proceedings of IJCAI-99 Workshop on Team Behaviour and Plan Recognition*, pages 53–63, 1999.

65. C. Heinze and L. Sterling. Using the UML to model knowledge in agent systems. In *AAMAS 2002 Poster Paper*, Bologna, Italy, 2002.

66. Clint Heinze. Intention oriented analysis: Toward a methodology for agent oriented software engineering. In *In Proceedings of the Fifth Australasian Cognitive Science Conference*, 2002.

67. Clint Heinze. *Modelling Intention Recognition for Intelligent Agent Systems*. PhD thesis, The University of Melbourne, Melbourne, Australia, 2003.

68. Clint Heinze, Martin Cross, Simon Goss, Torgny Josefsson, Ian Lloyd, Graeme Murray, Michael Papasimeon, and Michael Turner. Agents of change: The impact of intelligent agent technology on the analysis of air operations. In N. Ichalkaranje Jain and G. Tonfoni, editors, *Advances in Intelligent Systems for Defence*, volume 2 of *Series on Innovative Intelligence*, pages Chapter 6, pages 229–264. World Scientific, River Edge, New Jersey, USA, 1st edition edition, 2002.

69. Clint Heinze, Michael Papasimeon, and Simon Goss. Issues in modelling sensor fusion in agent based simulation of air operations. In *In Proceedings of the Sixth International Conference on Information Fusion*, volume 1, pages 296–301, Cairns, Australia, 2003. This paper was withdrawn at the last minute due to our inability to attend so although it was accepted to the conference it was never presented.

70. Clint Heinze, Michael Papasimeon, Simon Goss, Martin Cross, and Russell Connell. Simulating fighter pilots. In Simon G. Thompson Michal Pěchouček and Holger Voos, editors, *Defence Industry Applications of Autonomous Agents and Multi-Agent Systems*, Whitestein Series in Software Agent Technologies and Autonomic Computing. Birkhäuser Basel, 2008.

71. Clint Heinze, Bradley Smith, and Martin Cross. Thinking quickly: Agents for modeling air warfare. In *Proceedings of the 9th Australian Joint Conference on Artificial Intelligence (AI' 98)*, pages 47–58, Brisbane, Australia, 1998.

72. Clinton Heinze, Adrian R. Pearce, Leon Sterling, and Simon Goss. Designing perception modules to shape information for agents. In *Revised Papers from the PRICAI 2000 Workshop Reader, Four Workshops held at PRICAI 2000 on Advances in Artificial Intelligence*, pages 239–248, London, UK, 2001. Springer-Verlag.

73. Alexander Helleboogh, Giuseppe Vizzari, Adeline Uhrmacher, and Fabien Michel. Modeling dynamic environments in multi-agent simulation. *Autonomous Agents and Multi-Agent Systems*, 14(1):87–116, February 2007.

74. Nick Howden, Ralph Ronnquist, Andrew Hodgson, and Andrew Lucas. JACK intelligent agents - summary of an agent infrastructure. In *5th International Conference on Autonomous Agents*, 2001.

75. Damian Isla and Bruce Blumberg. New challenges for character-based AI for games. In *Proceedings of the AAAI Spring Symposium on AI and Interactive Entertainment*, Palo Alto, California, March 2002.

76. Nicholas R. Jennings. On agent-based software engineering. *Artificial Intelligence*, 177(2):277–296, 2000.

77. Keith S. Jones. Special issue: How shall affordances be refined? four perspectives. *Journal of Ecological Psychology*, 15(2), 2003.

78. Keith S. Jones. What is an affordance? *Journal of Ecological Psychology: Special Issue: How Shall Affordances Be Refined? Four Perspectives*, 15(2):107–114, 2003.

79. R. M. Jones, J. E. Laird, P. E. Nielsen, K. J. Coulter, P Kenny, and F. V. Koss. Automated intelligent pilots for combat flight simulation. *AI Magazine*, 20(1):27–41, 1999.

80. Troy Jordan, Martin Raubal, Bryce Gartrell, and Max J. Egenhofer. An affordance-based model of place in GIS. In T. Poiker and N. Chrisman, editors, *Eighth International Symposium on Spatial Data Handling*, pages 98–109, Vancouver, Canada, 1998.

81. Thomas Juan, Adrian Pearce, and Leon Sterling. ROADMAP: Extending the GAIA methodology for complex open systems. In *First International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 3–10. ACM Press, 2002.

82. Thomas Juan, Leon Sterling, and Michael Winikoff. Assembling agent oriented software engineering methodologies from features. In *Proceedings of the Third International Workshop on Agent-Oriented Software Engineering*, 2002.

83. D. Kinny, M. Georgeff, and A. Rao. A methodology and modelling technique for systems of BDI agents. In *Proceedings of the Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW'96)*, 1996.

84. Alex Kirlik. On Stoffregen's definition of affordances. *Journal of Ecological Psychology*, 16(1):73–77, 2004.

85. Franziska Klügl, Manuel Fehler, and Rainer Herrler. About the role of the environment in multi-agent simulations. In Danny Weyns, H. Van Dyke Parunak, and Michel Fabien, editors, *Environments for Multi-Agent Systems: First International Workshop on Environments for Multi-Agent Systems (E4MAS 2004)*, volume 3374 of *Lecture Notes in Computer Science (LNCS)*, pages 127–149, New York, NY, USA, February 2005. Springer-Verlag, Berlin.

86. Michael Lees, Brian Logan, Rob Minson, Ton Oguara, and Georgios Theodoropoulos. Modelling environments for distributed simulation. In Danny Weyns, H. Van Dyke Parunak, and Michel Fabien, editors, *Environments for Multi-Agent Systems: First International Workshop on Environments for Multi-Agent Systems (E4MAS 2004)*, volume 3374 of *Lecture Notes in Computer Science (LNCS)*, pages 150–167, New York, NY, USA, February 2005. Springer-Verlag, Berlin.

87. Lars Liden. Using nodes to develop strategies for combat with multiple enemies. In *2001 AAAI Spring Symposium on Artificial Intelligence and Interactive Entertainment.* AAAI Press, 2001.

88. William M. Mace. *Special Issue : How are affordances related to events? An exchange of views*, volume 12, Number 1 of *Journal of Ecological Psychology.* Lawrence Erlbaum and Associates, Inc., 2000.

89. William M. Mace. Special issue: How are affordances related to events? an exchange of views. *Journal of Ecological Psychology*, 12(1), 2000.

90. R. McAlinden and W. Clevenger. A culturally-enhanced environmental framework for virtual environments. In *Proceedings of Behavior Representation in Modeling and Simulation (BRIMS)*, Baltimore, Maryland, 2006.

91. Ryan McAlinden, Michael van Lent, William Clevenger, and Wen Ten. Using environmental annotations and affordances to model culture. In *Artificial Intelligence and Interactive Digital Entertainment Conference Demonstrations*, Marina del Rey, CA, USA, June 2006.

92. D. McIlroy and C. Heinze. Advanced operational reasoning for Hornet tactics analysis. In *Proceedings of the Second International Simulation Technology and Training Conference (SimTect '97)*, Canberra, Australia, 1997.

93. D McIlroy, C Heinze, D Appla, P Busetta, G Tidhar, and A Rao. Towards credible computer-generated forces. In S. Sestito, P. Beckett, G. Tudor, and T.J. Triggs, editors, *Second International Simulation Technology and Training Conference (SimTect '97)*, pages 234–239, Melbourne, Australia, 1997.

94. D. McIlroy, B. Smith, C. Heinze, and M. Turner. Air defence operational analysis using the SWARMM model. In *Proceedings of the Asia Pacific Operations Research Symposium (APORS'97)*, Melbourne, Australia, 1997.

95. David McIlroy and Clint Heinze. Air combat tactics in the smart whole air mission model (SWARMM). In *First Simulation Training and Technology Conference (SimTect)*, Melbourne, Australia, March 1996.

96. Steven Garrett Michael F. Young, Andrew Depalma. Situations, interaction, process and affordances: An ecological psychology perspective. *Journal of Instructional Science*, 30:47–63, 2002.

97. Claire F. Michaels. Affordances: Four points of debate. *Journal of Ecological Psychology*, 15(2):135–148, 2003.

98. Claire F. Michaels and Claudia Carello. *Direct Perception.* Century Psychology Series. Prentice Hall, 1981.

99. A. Nareyek. Intelligent agents for computer games. In *Second International Conference on Computers and Games (CG 2000)*, pages 414–422, 2000.

100. E. Norling and C. Heinze. Naturalistic decision making and agent oriented cognitive modelling. In *Proceedings of Fifth Australasian Cognitive Science Conference*, 2000.

101. Donald A. Norman. Cognition in the head and in the world: An introduction to the special issue on situated action. *Cognitive Science*, 17(1):1–6, 1993.

102. Donald A. Norman. Cognition in the head and in the world: Special issue on situated action. *Journal of Cognitive Science*, 17, 1993.

103. Donald A. Norman. Affordances, conventions and design. *Interactions*, 6(3), May 1999.

104. Donald A. Norman. *The Design of Everyday Things*. Basic Books, 2002.

105. J. Odell, H. Van Dyke Parunak, and Bernhard Bauer. Extending UML for agents. In *Proceedings of AOIS Workshop AAAI*, 2000.

106. James Odell, H. Van Dyke Parunak, Mitch Fleishcer, and Sven Breucker. Modeling agents and their environment. In *Proceedings of the 4th International Workshop on Agent Oriented Software Engineering (AOSE '02)*, Bologna, Italy, 2002.

107. Fabio Y. Okuyama, Rafael H. Bordini, and Antônio Carlos da Rocha Costa. ELMS: An environment description language for multi-agent simulation. In Danny Weyns, H. Van Dyke Parunak, and Michel Fabien, editors, *Environments for Multi-Agent Systems: First International Workshop on Environments for Multi-Agent Systems (E4MAS 2004)*, volume 3374 of *Lecture Notes in Computer Science (LNCS)*, pages 91–108, New York, NY, USA, February 2005. Springer-Verlag, Berlin.

108. Andrea Omicini, Alessandro Ricci, and Mirko Viroli. Coordination artifacts as first-class abstractions for mas engineering: State of research. In *Software Engineering for Multi-Agent Systems IV*, volume 3914/2006 of *Lecture Notes in Computer Science (LNCS)*, pages 71–90. Springer, Heidelberg, Germany, April 2006.

109. Jeff Orkin. 12 tips from the trenches. In Steve Rabin, editor, *AI Game Programming Wisdom*, pages 29–36. Charles River Media, Hingham, MA, USA, 2002.

110. G. A. Papadopoulos and F. Arbab. Coordination models and languages. *Advances in Computers*, 46:329–400, 1998.

111. Michael Papasimeon. Intelligent environments for agents. In *2002 Australian Cognitive Science Conference (OzCogSci 2002)*, Fremantle, Western Australia, 2002.

112. Michael Papasimeon and Clint Heinze. Extending the UML for designing jack agents. In *Proceedings of the Australian Software Engineering Conference (ASWEC 2001)*, pages 89–97, Canberra, Australia, 2001.

113. Michael Papasimeon, Adrian R. Pearce, and Simon Goss. The human agent virtual environment. In *AAMAS '07: Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, pages 1–8, New York, NY, USA, 2007. ACM.

114. Richard W. Pew and Anne S. Mavor, editors. *Modeling Human and Organizational Behavior: Applications to Military Simulation.* National Research Council. National Academy Press, 2000.

115. Eric Platon, Marco Namei, Nicolas Sabouret, Shinichi Honiden, and H. Van Dyke Parunak. Mechanisms for environments in multi-agent systems: Survey and opportunities. *Autonomous Agents and Multi-Agent Systems*, 14(1):31–48, February 2007.

116. J. S. Przemieniecki. *Mathematical Methods in Defense Analyses.* American Insitute of Aeronautics and Astronautics, Reston, Virginia, USA, 3rd edition edition, 2000.

117. Murphy R. R. Case studies of applying Gibson's ecological approach to mobile robots. *IEEE Transactions on Systems, Man and Cybernetics*, 1999.

118. Steve Rabin. Designing a general robust AI engine. In Mark DeLoura, editor, *Game Programming Gems*, chapter 3.0, pages 221 – 236. Charles River Media, Rockland, Massachusetts, USA, 2000.

119. Steve Rabin. Promising game AI techniques. In Steve Rabin, editor, *AI Game Programming Wisdom 2.* Charles River Media, Hingham, Massachusetts, USA, 2004.

120. Anand S. Rao and Michael P. Georgeff. Modeling rational agents within a BDI-architecture. In James Allen, Richard Fikes, and Erik Sandewall, editors, *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning*, pages 473–484. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA, 1991.

121. Anand S. Rao and Michael P. Georgeff. BDI agents: from theory to practice. In *Proceedings of the First International Conference on Multiagent Systems*, San Fransisco, 1995.

122. Anand S. Rao and Graeme Murray. Multi-agent mental-state recognition and its application to air-combat modelling. In *Proceedings of the 13th International Workshop on Distributed Artificial Intelligence (DAI-94)*, pages 283–304, Seattle, Washington, USA, 1994.

123. M. Raubal. Ontology and epistemology for agent-based wayfinding simulation. *International Journal of Geographical Information Science*, 15(7):653–665, 2001.

124. Martin Raubal. *Agent-Based Simulation of Human Wayfinding: A Perceptual Model for Unfamiliar Buildings.* PhD thesis, Technical University Vienna, 2001.

125. Martin Raubal. Ontology and epistemology for agent-based wayfinding simulation. *International Journal of Geographical Information Science*, 15(7):653–665, 2001.

126. A. Ricci, M. Viroli, and A. Omicini. Programming mas with artifacts. In *Workshop on Programming Languages for Multi-Agent Systems (PROMAS), 4th International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'05)*, Utrecht, The Netherlands, 2005.

127. Alessandro Ricci and Mirko Viroli. Coordination artifacts: A unifying abstraction for engineering environment-mediated coordination in mas. *Informatica*, 29(4):433–443, November 2005.

128. Michael A. Riley and Marie-Vee Santana. Mutuality relations, observations, and intentional constraint. *Ecological Psychology: Special Issue: How are Affordances Related to Events? An Exchange of Views*, 12(1):79–86, 2000.

129. Erich Rome, Joachim Hertzberg, Georg Dorffner, and P. Doherty. Towards affordance-based robot control. In Erich Rome, Joachim Hertzberg, and Georg Dorffner, editors, *Lecture Notes in Computer Science*, volume 4760. Springer-Verlag, Dagstuhl Castle, Jermany, June 2006 2008.

130. Paul S. Rosenbloom, John E. Laird, and Allen Newell, editors. *The Soar Papers (Volume 1): Research on Integrated Intelligence.* MIT Press, Cambridge, MA, USA, 1993.

131. Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach.* Prentice Hall, 1995.

132. E. Sahin, M. Cakmak, M.R. Dogar, E. Ugur, and G. Ucoluk. To afford or not to afford: A new formalization of affordances towards affordance-based robot control. *Adaptive Behavior*, 15(4):447–472, 2007.

133. Colleen M. Seifert. Situated cognition. In Robert A. Wilson and Frank C. Keil, editors, *The MIT Encyclopedia of the Cognitive Sciences.* The MIT Press, Cambridge, Massachussetts, 1999.

134. Pedro Sequira, Marco Vala, and Ana Paiva. ”What can I do with this?” Finding possible interactions between characters and objects. In *Autonomous Agents and Multi-Agent Systems (AAMAS) Conference*, Honolulu, Hawaii, May 2007.

135. Robert L. Shaw. *Fighter Combat: Tactics and Maneuvering.* Naval Institute Press, Anapolis, Maryland, USA, 1985.

136. Barry G. Silverman. Towards realism in human performance simulation. In James W. Ness, Darren R. Ritzer, and Victoria Tepe, editors, *The Science and Simulation of Human Performance*, pages 469–498. Elsevier, 2004.

137. Barry G. Silverman, Norman I. Badler, Nuria Pelechano, and Kevin O’Brien. Crowd simulation incorporating agent psychological models, roles and communication. In *First International Workshop on Crowd Simulation (V-CROWS ’05)*, 2005.

138. Jake Simpson. Scripting and Sims 2: Coding the psychology of little people. In *Game Developers Conference (GDC 2005)*, San Fransisco, USA, 2005.

139. Liz Sonenberg and Gil Tidhar. Observations on team-oriented mental state recognition. In *Proceedings of the 1999 IJCAI Workshop on Team Modelling and Plan Recognition*, 1999.

140. J. Michael Spivey. *The Z Notation: A Reference Manual.* Prentice Hall, 1992.

141. Renee Steiner, Gary Leask, and Rym Z. Mili. An architecture for MAS simulation environments. In *Environments for Multi-Agent Systems II: Second International Workshop on Environments for Multi-Agent Systems (E4MAS 2005)*, volume 3830 of *Lecture Notes in Computer Science (LNCS)*, pages 50–67, Utrecht, The Netherlands, February 2006. Springer-Verlag, Berlin.

142. Thomas A. Stoffregen. Affordances and events. *Ecological Psychology: Special Issue: How are Affordances Related to Events? An Exchange of Views*, 12(1):1–28, 2000.

143. Thomas A. Stoffregen. Affordances and events. *Journal of Ecological Psychology: Special Issue: How are Affordances Related to Events? An Exchange of Views*, 12(1):1–28, 2000.

144. Arnon Sturm. A framework for evaluating agent-oriented methodologies. In *Proceedings of the International Bi-Conference Workshop on Agent Oriented Information Systems (AOIS)*, volume 3030 of *Lecture Notes in Computer Science (LNCS)*, pages 94–109. Springer, 2003.

145. Lucy Suchman. Response to Vera and Simon's situated action: A symbolic interpretation. *Cognitive Science*, 17(1):71–75, 1993.

146. Lucy A. Suchman. *Plans and Situated Actions : The Problem of Human-Machine Communication.* Cambridge University Press, 1987.

147. M. Tambe. Agent architectures for flexible, practical teamwork. In *Proceedings of National Conference on Artificial Intelligence (AAAI-97)*, 1997.

148. Miland Tambe, W. J. Johnson, R. M. Jones, K. Koss, J. E. Laird, P. S. Rosenbloom, and K. Scwamb. Intelligent agents for interactive simulation environments. *AI Magazine*, Spring:15–39, 1995.

149. Dale Thomas. New paradigms in artificial intelligence. In Steve Rabin, editor, *AI Game Programming Wisdom 2.* Charles River Media, 2004.

150. G. Tidhar, P. Busetta, and C. Heinze. Intention recognition in air combat. Technical report, Defence Science and Technology Organisation (DSTO), 1999.

151. G Tidhar, C Heinze, S Goss, G Murray, D Appla, and I Lloyd. Using intelligent agents in military simulation or "using agents intelligently". In *Eleventh Innovative Applications of Artificial Intelligence Conference*, pages 829–836, Menlo Park, California, USA, 1999. American Association for Artificial Intelligence.

152. G Tidhar, C Heinze, and M Selvestrel. Flying together: Modelling air mission teams. *Applied Intelligence*, 8(3):195–218, 1998.

153. G Tidhar, G Murray, and S Steuart. Computer-generated forces and agent technology. In *Australian Joint Conference on Artificial Intelligence, Workshop on AI in Defense*, Canberra, Australia, July 1995.

154. G. Tidhar, M. Selvestrel, and C. Heinze. Modelling teams and team tactics in whole air mission modelling. In G. F. Forsyth and M. Ali, editors, *Proceedings of the Eighth International Conference on Industrial and Engineering Applications of Artificial Intelligence (IEA/AIE-95)*, pages 373–381, Melbourne, Australia, June 1995.

155. Gil Tidhar. *Organisation-Oriented Systems: Theory and Practice.* PhD thesis, The University of Melbourne, Melbourne, Australia, 1999.

156. Ivan Trencasnsky and Radovan Cervenka. Agent modeling language (AML): A comprehensive approach to modeling mas. *Informatica*, 29(4):391–400, November 2005.

157. TTCP. *TTCP Guide for Understanding and Implementing Defense Experimentation (GUIDEx).* The Technical Cooperation Program (TTCP), Ottawa, Canada, 2006.

158. M. T. Turvey. Affordances and prospective control: An outline of the ontology. *Journal of Ecological Psychology*, 4:173–187, 1992.

159. M. T. Turvey and T. E. Shaw. Toward and ecological physics and physical psychology. In R. L. Solso and D. W. Massaro, editors, *The Science of the Mind*. Oxford University Press, 1995.

160. Paul Valckenaers, John Sauter, Carles Sierra, and Juan Antonio Rodriguez-Aguilar. Applications and environments for multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 14(1):61–86, February 2007.

161. Jean Paul van Waveren. The Quake III Arena Bot. Master's thesis, Delft University of Technology, Delft, The Netherlands, 2001.

162. Alonso H. Vera and Herbert A. Simon. Situated action: A symbolic interpretation. *Cognitive Science*, 17(1):7–48, 1993.

163. Alonso H. Vera and Herbert A. Simon. Situated action: Reply to reviewers. *Cognitive Science*, 17(1):77–86, 1993.

164. Alonso H. Vera and Herbert A. Simon. Situated action: Reply to William Clancey. *Cognitive Science*, 17(1):117–133, 1993.

165. M Viezzer and C. M. Nieuwenhuis. Learning affordance concepts: some seminal ideas. In *In Proceedings of the Workshop on Modeling Natural Action Selection at the 2005 International Joint Conference on Artificial Intelligence (IJCAI05)*, Edinburgh, Scotland, 2005.

166. M. Viroli, A. Omicini, and A. Ricci. Engineering MAS environments with artifacts. In D. Weyns, H. Van Dyke Parunak, and F. Michel, editors, *2nd International Workshop on Environments for Multi-Agent Systems (E4MAS 2005)*, Utrecht, The Netherlands, July 2005.

167. Marko Viroli and Alessandro Ricci. Tuple-based coordination models in event-based scenarios. In *Proceedings of the 22nd International Conference on Distributed Computing Systems Workshops (ICDCSW'02)*. IEEE Computer Society, 2002.

168. Mirko Virolo, Tom Holvoet, Alessandro Ricci, Kurt Schelfthout, and Franco Zambonelli. Infrastructures for the environment of multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 14(1):49–60, February 2007.

169. John A Warden. *The Air Campaign.* toExcel Press, San Jose, California, USA, 2000.

170. Mark Weiser and John S. Brown. Designing calm technology. *PowerGrid Journal v 1.01*, 1996.

171. D. Weyns, H. Parunak, F. Michel, T. Holvoet, and J. Ferber. Environments for multiagent systems: State-of-the-art and research challenges. In Danny Weyns, H. Van Dyke Parunak, and Michel Fabien, editors, *Environments for Multi-Agent*

Systems : Proceedings of the First International Workshop on Environments for Multi-Agent Systems (E4MAS 2004), volume 3374 of *Lecture Notes in Artificial Intelligence (LNAI)*, pages 1–47, New York, NY, USA, 2005. Springer-Verlag, Berlin, Germany.

172. Danny Weyns and Tom Holvoet. On the role of environments in multiagent systems. *Informatica*, 29(4):409–421, November 2005.

173. Danny Weyns, Andrea Omicini, and James Odell. Environments as a first class abstraction in multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 14(1):5–30, February 2007.

174. Danny Weyns, Giuseppe Vizzari, and Tom Holvoet. Environments for situated multi-agent systems: Beyond infrastructure. In Danny Weyns, H. Van Dyke Parunak, and Michel Fabien, editors, *Environments for Multi-Agent Systems II: Second International Workshop on Environments for Multi-Agent Systems (E4MAS 2005)*, volume 3830 of *Lecture Notes in Computer Science (LNCS)*, pages 1–17, Utrecht, The Netherlands, February 2006. Springer-Verlag, Berlin.

175. Jim Woodcock and Jim Davies. *Using Z: Specification, Refinement and Proof.* Prentice-Hall, 1996.

176. M. Wooldridge. Agent-based software engineering. In *IEEE Proceedings on Software Engineering*, pages 22–37, 1997.

177. M. Wooldridge, N. R. Jennings, and D. Kinny. The GAIA methodology for agent-oriented analysis and design. *Autonomous Agents and Multi-Agent Systems*, 3(3):285–312, 2000.

178. Michael Wooldridge. *Reasoning About Rational Agents.* MIT Press, Cambridge, Massachusetts, 2000.

179. Michael Wooldridge. *An Introduction to MultiAgent Systems.* John Wiley and Sons Ltd., Chichester, England, 2002.

180. Franco Zambonelli and Andrea Omicini. Challenges and research directions in agent-oriented software engineering. *Autonomous Agents and Multi-Agent Systems*, 9(3):253–284, February 2004.

181. Jiajie Zhang and Vimla L. Patel. Distributed cognition, representation and affordance. *Pragmatics and Cognition*, 14(2):333–341, 2006. Distributed Cognition: Special Issue.

Page classification: UNCLASSIFIED

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION DOCUMENT CONTROL DATA	1. CAVEAT/PRIVACY MARKING

2. TITLE Modelling Agent-Environment Interaction in Multi-Agent Simulations with Affordances	3. SECURITY CLASSIFICATION Document (U) Title (U) Abstract (U)
4. AUTHOR Michael Papasimeon	5. CORPORATE AUTHOR Defence Science and Technology Organisation 506 Lorimer St, Fishermans Bend, Victoria 3207, Australia

6a. DSTO NUMBER DSTO–RR–0349	6b. AR NUMBER	6c. TYPE OF REPORT Research Report	7. DOCUMENT DATE April, 2010

8. FILE NUMBER 2009/1169974/1	9. TASK NUMBER LRR 07/245	10. SPONSOR CAOD	11. No. OF PAGES 224	12. No. OF REFS 181

13. URL OF ELECTRONIC VERSION http://www.dsto.defence.gov.au/corporate/reports/DSTO–RR–0349.pdf	14. RELEASE AUTHORITY Chief, Air Operations Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved for Public Release

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SOUTH AUSTRALIA 5111

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CITATION IN OTHER DOCUMENTS

No Limitations

18. DSTO RESEARCH LIBRARY THESAURUS

Agents	Artifical Intelligence
BDI Agents	Cognitive Science
Psychology	Visual Perception
Flight Simulation	Software Engineering
Air Operations	

19. ABSTRACT

A computational model of agent-environment interaction is presented that has been inspired by the theory of affordances from ecological psychology. The model is developed in the context of agent-environment interaction in multi-agent simulation in a virtual environment. The research is motivated by multi-agent simulation of military operations, where the agents represent cognitive models of military operators. A mechanism for evaluating the model is provided through a number of design and implementation interpretations in two distinct large scale multi-agent simulation systems.

Page classification: UNCLASSIFIED